Dialog Navigator: A Navigation System from Vague Questions to Specific Answers based on Real-World Text Collections

Yoji Kiyota

Abstract

As computers and their networks continue to be developed, our day-to-day lives are being surrounded by increasingly more complex instruments, and we often have to ask questions about using them. At the same time, large collections of texts to answer these questions are being gathered. Therefore, there are potential answers to many of our questions that exist as texts somewhere. However, there are various gaps between our various questions and the texts, and these prevent us from accessing appropriate texts to answer our questions. The gaps are mainly composed of both *expression* and *vagueness* gaps. When we seek texts for answers using conventional keyword-based text retrieval systems, we often have trouble locating them. In contrast, when we ask experts on instruments or operators of call centers, they can resolve the various gaps, by interpreting our questions flexibly, and by producing some ask-backs.

The problem with experts and call centers is that they are not always available. Two approaches have been studied to resolve the various gaps: the extension of keyword-based text retrieval systems, and the application of artificial intelligence techniques. However, these approaches have their respective limitations. The former uses texts or keywords as methods for ask-back questions, but these methods are not always suitable. The latter requires a specialized knowledge base described in formal languages, so it cannot be applied to existing collections with large amount of texts.

This thesis targets real-world the large text collections provided by Microsoft Corporation, and addresses a novel methodology to resolve the gaps between various user questions and the texts. The methodology consists of two key solutions: precise and flexible methods of matching user questions with texts based on NLP (natural language processing) techniques, and ask-back methods using the matching methods. First, the matching methods, including sentence structure analysis and expression gap resolution, are described. In addition, these methods are extended into matching through metonymy, which is frequently observed in natural languages. After that, a solution to make ask-

backs based on these matching methods, by using two kinds of ask-backs that complement each other, is proposed. Both ask-backs navigate users from vague questions to specific answers. Finally, our methodology is evaluated through the real-world operation of a dialog system, *Dialog Navigator,* in which all the proposed methods are implemented.

Chapter 1 discusses issues on information retrieval, and present which issues are to be solved. That is, it examines the question logs from a real-world natural-language-based text retrieval system, and organizes types and factors of the gaps. The examination indicates that some gaps between user questions and texts cannot be resolved well by methods used in previous studies, and suggests that both interactions with users and applicability to real-world text collections are needed. Based on the discussion, a solution to deal with these gaps is proposed, by advancing an approach employed in open-domain question-answering systems, *i.e.,* utilization of recent NLP techniques, into resolving the various gaps.

Chapter 2 proposes several methods of matching user questions with texts, based on the NLP techniques. Of these techniques, sentence structure analysis through full-parsing is essential for two reasons: first, it enables expression gaps to be resolved beyond the keyword level; second, it is indispensable in resolving vagueness gaps by providing ask-backs. Our methods include: sentence structure analysis using a Japanese parser KNP, expression-gap resolution based on two kinds of dictionaries, text-collection selection through question-type estimates, and score calculations based on sentence structures. An experimental evaluation on testsets shows significant improvements of performance by our methods.

Chapter 3 proposes a novel method of processing metonymy, as an extension of the matching methods proposed in Chapter 2. Metonymy is a figure of speech in which the name of one thing is substituted for that of something else to which it is related, and this frequently occurs in both user questions and texts. Namely, this chapter addresses the automatic acquisition of pairs of metonymic expressions and their interpretative expressions from large corpora, and applies the acquired pairs to resolving structural gaps caused by metonymy. Unlike previous studies on metonymy, the method targets both recognition and interpretation process of metonymy. The method acquired 1,126 pairs from corpora, and over 80% of the pairs were correct as interpretations of metonymy. Furthermore, an experimental evaluation on the testsets demonstrated that introducing the acquired pairs significantly improves matching.

Chapter 4 presents a strategy of navigating users from vague questions to specific texts based on the previously discussed matching methods. Of course, it is necessary to make some use of ask-backs to achieve this, and this strategy involves two approaches: *description extraction* as a bottom-up approach, and *dialog cards* as a top-down approach. The former extracts the neighborhoods of the part that matches the user question in each text through matching methods. Such neighborhoods are mostly suitable for ask-backs that clarify vague user questions. However, if a user's question is too vague, this approach often fails. The latter covers vague questions based on the know-how of the call center; dialog cards systematize procedures for ask-backs to clarify frequently asked questions that are vague. Matching methods are also applied to match user questions with the cards. Finally, a comparison of the approaches with those used in other related work demonstrates the novelty of the approaches.

Chapter 5 describes the architecture for *Dialog Navigator,* a dialog system in which all the proposed methods are implemented. The system uses the real-world large text collections provided by Microsoft Corporation, and it has been open to the public on a website from April 2002. The methods were evaluated based on the real-world operational results of the system, because the various gaps to be resolved should reflect those in the real-world. The evaluation proved the effectiveness of the methods: more than 70% of all user questions were answered with relevant texts, the behaviors of both users and the system were reasonable with most dialogs, and most of the extracted descriptions for ask-backs were suitably matched.

Chapter 6 concludes the thesis.

Acknowledgments

I would like to express my gratitude to Professor Takashi Matsuyama for his supervision of this thesis and for his constructive suggestions.

I am deeply grateful to Associate Professor Satoshi Sato for his supervision and fruitful suggestions to complete this thesis. I am also grateful to Professor Tatsuya Kawahara, who gave a lot of valuable advice and comments, and took me in his laboratory as a postdoctoral fellow.

I would like to express my sincere appreciation to Associate Professor Sadao Kurohashi of University of Tokyo for his constant supervision, invaluable suggestions, and continuous encouragements to complete both master's and this thesis. He guided me in the right direction when necessary, both personally and professionally.

I am profoundly grateful to Professor Makoto Nagao, the President of NICT, who supervised my bachelor's thesis and introduced me to the joys of research on natural language processing. I am also grateful to Professor Jun'ichi Nakamura, who supervised my study and gave me a lot of valuable advice when I was an undergraduate and graduate student, and unfortunately passed away in 2001.

I am deeply indebted to Professor Toyoaki Nishida, who took me in his laboratory as a research student at University of Tokyo and gave me thoughtful and valuable advice. I am also grateful to Professor Jun'ichi Tsujii, Professor Hitoshi Nakagawa, and Associate Professor Kumiko Tanaka-Ishii for constructive and helpful advice when I was at University of Tokyo.

I owe a great deal to all previous and current members of Language Media Laboratory (formerly Professor Nagao's Laboratory) at Kyoto University, and those of Professor Nishida's and Associate Professor Kurohashi's Laboratory at University of Tokyo. Especially, I wish to thank Dr. Masaki Murata (currently at NICT), Mr. Daisuke Kawahara (currently at University of Tokyo), and Mr. Masatoshi Tsuchiya (currently at Toyohashi University of Technology) for helpful suggestions and generous instruction in basic com-

puter skills. I also wish to thank Dr. Masashi Okamoto (currently at University of Tokyo), who gave me a lot of helpful and interesting advice from a linguistic standpoint. I would like to thank all other members of both laboratories for their helpful supports and fruitful discussions.

I am grateful to the members of Professor Kawahara's Laboratory (formerly Speech Media Laboratory) for helpful supports and valuable discussions. I especially would like to thank Dr. Kazunori Komatani (currently at Professor Okuno's Laboratory) and Mr. Teruhisa Misu, who have been working together with me for the speech interface for Dialog Navigator.

I am thankful to people at Microsoft Co., Ltd. for their enormous supports on finance and resources. Especially, I wish to thank Ms. Fuyuko Kido for her considerable devotion to the continual operation of Dialog Navigator.

I also would like to thank Professor Hayato Yamana and his students at Waseda University for their contribution to the operation of Dialog Navigator.

I am grateful to Mr. Takashi Tachibana, who gave me a lot of invaluable comments, based on his precise and insightful analysis of the numerous question logs for Dialog Navigator.

Finally, I wish to thank my family and friends for their continuous supports and encouragements.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recently, as sophisticated electronic products have become more popular, numerous questions have been raised on using such products in various situations: we are often perplexed by mysterious error messages appearing on personal computers, Linux seems mumbo-jumbo for beginners, the new functions of a cellular phone are not that easy to master, and setting DVD video recorders to record while we are away needs complex manipulations. Such products are usually equipped with manuals, but these are not always helpful. In some cases, they contain too much information, and in other cases, they contain many esoteric technical terms.

Meanwhile, as the capacity of computer storage increases, and as computer networks continue to grow, large collections of texts to answer such questions have been accumulated. Manufacturers of products have call centers, and they have information on frequently asked questions and answers to these on large databases. Some heavy users of products have websites, and they provide tips on use accessible to the public. This means that there is probably an answer to any question somewhere.

However, we often have trouble finding the appropriate texts that will answer our questions on using complex products. Consider the case where we are seeking such texts using text retrieval systems (*e.g.* web search engines). First, we often have difficulty in selecting suitable keywords to explain our situations, because we are usually unfamiliar with the technical terms related to the products (Conversely, if we are familiar with such terms, *i.e.,* if we were versed in their products, we seldom have questions about them). Even if we manage to select a keyword, few appropriate texts can be located: in some cases, there are no relevant texts, in others, there is an impossible number causing us to give up our search.

What prevents us from finding appropriate texts to answer our questions? The main difficulty lies in the frequent gaps between our questions and the texts, such as the gaps in expression, vagueness, and belief. Usually, contemporary keyword-based text retrieval systems (including web search engines) have no capabilities for resolving these gaps.

In contrast, the operators of call centers and experts can resolve these *interactively*. For example, if we asked the vague question "An error occurred", they resolve the gaps through ask-backs: *e.g.* "What was the error message?", "How did it occur?", or "Which version are you using?" Also, even if we had asked inaccurate questions, they could interpret these *flexibly*.

The problem with call centers and experts is that they are not always available. In addition, the operation costs of maintaining call centers create big problems for manufacturers. That is why we need a system that can resolve gaps between our questions and the answers through interactions, based on existing large text collections.

This thesis addresses *Dialog Navigator*, a system which has the capability of resolving these gaps through dialogs based on real-world text collections. To achieve this system, two key solutions based on NLP (natural language processing) techniques are proposed: precise and flexible matching of user questions with texts, and ask-backs based on matching.

1.1 Previous Work on Information Retrieval

This section gives an overview of previous work on information retrieval in a broad sense, including text retrieval and QA (question-answering). First, basic studies on text retrieval systems and QA systems are described. After that, the contributions NLP techniques have made to information retrieval are summarized.

1.1.1 Basic Studies

Studies on information retrieval systems began at the end of the 1950s. Luhn proposed some of the fundamental ideas, *e.g.* automatic keyword extraction [1]. He found that keywords that occurred at a medium frequency in a document were usually important to that document. Then, overwhelmed by the Sputnik shock in 1957, the U.S. government began to support research on systems to retrieve scientific information [2]. As a result, in the early to mid 1960s, several systems such as MEDLARS [3] were put into practical use.

In general, those systems were targeted at bibliographic information (*e.g.* titles, authors, and keywords).

With the increasing development of computer science in the 1960s, 1970s and 1980s, systems to retrieve abstracts and full texts were studied. MEDLARS was developed into MEDLINE (MEDLARS on-LINE) in 1971, and started services on retrieving abstracts for papers. Salton *et al.* developed SMART, a full text retrieval system which they had improved over three decades, and assessed several fundamental methods, including TF.IDF, the vector space model, keyword expansion, and relevance feedback [4]. About this time, several retrieval models, such as the probabilistic model, extended Boolean model, fuzzy set model, and network model were also proposed. Most systems automatically extracted keywords from each text, and approximated each text against a collection of keywords.

In the 1990s, workshops for evaluating text retrieval systems (*e.g.* TREC [5], CLEF [6], and NTCIR [7]) began to be held, and large test collections were constructed. Various systems were examined at the earlier workshops, including not only those that employed keyword-based methods, but also those that employed unique methods such as full-parsing-based methods. However, the outcome was that these innovative approaches did not perform as well as the keyword-based ones, and therefore participants have converged on keyword-based methods in recent workshops.

With the maturing of the Internet since the latter half of the 1990s, text retrieval systems have become popular as web search engines. These systems are also based on keyword-based methods.

As large text collections became available in 1990s, QA systems based on natural language text collections were actively studied. In the early years, some prototype systems such as FAQ Finder [8] were developed. After that, open-domain QA systems based on unstructured texts (*e.g.* newspapers and web pages) have been actively studied by TREC QA Track [5] and NTCIR QAC [9].

1.1.2 Application of NLP

Most studies on text retrieval systems have assumed that information in the real world can be described as texts written in natural language, and that user questions can also be described in natural language. To conform to both of these, text retrieval systems should convert both of them into a regular format. These systems have adopted natural language processing (NLP) techniques for this conversion.

However, from the 1960s, the contribution of NLP techniques to text retrieval systems has been limited to keyword extraction. Efforts on applying deeper NLP techniques to improve text retrieval have already been made on Salton's SMART system [10], but he reported that no significant improvements were observed. In 1996, an NLP track was held at TREC-5, and NLP techniques such as full-parsing were applied by some participants. The result was that the NLP techniques were useful for text retrieval, but their impact was limited compared with costs. Contemporary web search engines do not depend on NLP techniques, except for morphological analysis; the improvements to retrieval have mainly been achieved by proximity matching, and exploitation of HTML structures and hyperlinks.

In contrast, unlike text retrieval systems, QA systems have to detect exact answers from retrieved texts. To do this, recent QA systems have adopted deeper NLP techniques including sentence structure analysis based on full-parsing. Early text-based QA systems such as MURAX [11] and FAQ Finder [8] exploited case analysis. Also, most recent open-domain QA systems [12–14] have adopted sentence structure analysis and question type analysis to find the relevant answers.

From the viewpoint of gaps between user questions and texts, all the above methods have limitations; they assume that every user question is specific. If someone asks vague questions, these methods will not work well.

1.2 Various Gaps between User Questions and Answers

In most cases, a text which contains answer A to question Q has an important feature: the text also contains a part Q' which is similar to Q, and Q' exists near A. As a result, matching Q to Q' mostly leads to A being detected. This feature has been generally utilized by studies on information retrieval and question answering. However, as the author indicated at the beginning, there are various gaps that prevent us from accessing specific texts to answer our various questions. These gaps usually exist between Q and Q'. Moreover, they range at various levels, and are caused by various factors.

This section organizes the levels and the factors, reviews previous studies on resolving the gaps, and proposes a solution to deal with them.

1.2.1 Types of Gaps

The author preliminarily examined the question logs of a real-world natural-language-based text retrieval system operated by Microsoft Corporation, and found that there are many types of the gaps. These are:

1. *Expression gaps at the keyword level.*

   In many cases, a keyword in a user question does not match the keyword in the answer text, although both keywords indicate the same thing. The following types of expression gaps have mainly been observed at the keyword level:

   - Variant notations. *e.g.* "**メール**" (*mēru*, mail) - "**メイル**" (*meiru*, mail) - *"mail"*; "ダイ**ア**ルアップ" (*daiaru appu*, dial-up) - "ダイ**ヤ**ルアップ" (*diyaru appu*, dial-up); "アダプタ" (*adaputa*, adapter) - "アダプタ**ー**" (*adaputā*, adapter); "**余**る" (*amaru*, be left) - "**あま**る" (*amaru*, be left); "落**と**す" (*otosu*, drop) - "落す" (*otosu*, drop). Especially in Japanese, these are often caused by multiple notation systems (*e.g. hiragana, katakana, kanji*, and Roman characters) and declensions of *kana* ending systems.
   - Abbreviation. *e.g.* "**デジ**タル**カメ**ラ" (*dejitaru kamera*, digital camera) - "デジカメ" (*dejikame*, digital camera); "**取**扱**説**明書" (*toriatsukai setsumeisho*, manual) - "取説" (*torisetsu*, manual); "**I**nternet **E**xplorer" - "IE".
   - Imported words and native words. *e.g.* "レイアウト" (*reiauto*, imported from English 'layout') - "配置" (*haichi*, layout); "フォーム" (*fōmu*, imported from English 'form') - "書式" (*shoshiki*, form).
   - Synonyms. *e.g.* "障害" (*shōgai*, disorder) - "不具合" (*fuguai*, defect); "3 次元の" (*san-jigen-no*, three-dimensional) - "立体的な" (*rittaiteki-na*, spacial).
   - Hypernyms and hyponyms. *e.g.* "ワープロソフト" (*wāpuro sofuto*, word processor application) - *"Word"*; "開発ツール" (*kaihatsu tsūru*, development tool) - *"VisualC++"*.
   - Misspellings. *e.g. "Perl"* (correct) - *"Parl"* (wrong); *"XLS"* (correct) - *"XSL"* (wrong).

2. *Expression gaps beyond the keyword level.*

Expression gaps not only exist at the keyword level, but also beyond the keyword level. Consider an expression gap at the phrase level as follows:

(1.1) メールを 読む
mēru-wo *yomu*
mail-acc read
'read mail'

(1.2) メールを 受信する
mēru-wo *jushin-suru*
mail-acc receive-(do)
'receive mail'

These two expressions are nearly equivalent to each other in the personal computer domain. Note that naive methods by which "読む" (*yomu*, read) and "受信する" (*jushin-suru*, receive) are regarded as synonyms do not work well, because these two keywords would have different senses in other contexts.

Here is another expression gap at the phrase level:

(1.3) *GIF*を 表示する
GIF-wo *hyōji-suru*
GIF-acc display-(do)
'display a GIF'

(1.4) *GIF*の 画像を 表示する
GIF-no *gazō-wo* *hyōji-suru*
GIF-gen image-acc display-(do)
'display a GIF image'

We can regard (1.3) as metonymy, and (1.4) as its interpretation. Namely, the user uses *"GIF"* as a substitution for *"GIF*の画像*"* (*GIF-no gazō*, GIF image).

3. *Vagueness gaps.*

We tend to start with vague questions, not with specific ones. The preliminary examination of the question logs revealed that more than 30 percent of questions by users were very vague. Such questions are completely different from the answers due to vagueness. For example, many users started with the following questions:

(1.5) エラーが 起きた
erā-ga *oki-ta*
error-nom occur-(did)

'An error occurred.'

(1.6) 文字化けした
moji-bake-shita
character-corrupt-(did)
'Characters were displayed corruptly.'

(1.7) 変更を 保存したい
henkō-wo *hozon-shitai*
change-acc save-(want to)
'(I) want to save the changes.'

By contrast, every answer that corresponds to (1.5-1.7) is described by a specific situation. Usually, error messages are specified with answers to (1.5), and names of applications are specified with answers to (1.6) and (1.7).

Note that users' intentions for (1.5) and (1.6) are also vague: they only described the symptoms. In (1.5), the user wants to know how to solve the error message problem, but his/her intentions are not specified externally.

4. *Belief gaps.*

At times, users may ask questions based on incorrect beliefs.

(1.8) *Word*で 文書を *PDF*形式で 出力したい
Word-de *bunsho-wo* *PDF-keishiki-de* *shutsuryoku-shitai*
Word-ins document-acc PDF-format-ins output-(want to)
'(I) want to output a document in PDF format using Word.'

The user seems to believe that Microsoft Word has a function for outputting PDF files. However, Word does not actually have such a function, and Adobe Acrobat should be installed to output PDF files. Although there is a text that answers how to output PDF files using Adobe Acrobat, it does not match the user's beliefs.

Note that each type of gap intertwines with others, and is caused by various factors. For example, the expression gaps could be caused both by inherited variety in natural languages, and by the users' lack of knowledge.

1.2.2 Factors Responsible for Gaps

To resolve the various gaps between user questions and answer texts, we have to deal with the factors behind the gaps. The author thinks that these gaps are caused by the

following factors:

(i) *Inherited variety in natural languages.*

There are generally various ways of expressing the one thing in any natural language. This inevitably causes expression gaps both at and beyond the keyword level.

(ii) *Lack of knowledge about terms and expressions.*

When asking questions about a product, we tend to be unfamiliar with technical terms and special expressions concerning the product. In such cases, we usually manage to somehow express our questions using similes, metaphors, metonymies, or inaccurate expressions that our acquaintances often use. This results in expression gaps at various levels. We also often frame vague questions because we do not know what expressions will be specific to our situation.

(iii) *Lack of knowledge on background information.*

When we have questions while using a product, we tend to be unsure about the background to the question, including the detailed features the product offers, and the environment and conditions under which the problem occurred. That is, our knowledge structure on the product is initially unstable.

This mainly causes vagueness gaps. Furthermore, it sometimes leads to our misunderstanding, and causes belief gaps.

(iv) *Expectations of person answering.*

We have a bias toward reluctance. We often unintentionally ask short and incomplete questions, and expect that the person answering will interpret our questions flexibly, or he/she will ask us to clarify them. When we ask additional questions, we usually omit already specified information, and expect that he/she will interpret the question contextually.

This factor also causes many vagueness gaps. If the person answering has no capabilities for dealing with such incomplete questions, it may also lead to belief gaps.

Generally, studies on text retrieval systems and QA systems have taken the first factor (i) into account. Resolution of the expression gaps at the keyword level caused by the

inherited variety in natural languages has been studied. We should note that expression gaps beyond the keyword level can still not be resolved well.

In contrast, not much attention has been paid to the other factors (ii)-(iv), although these factors are critical for information retrieval. Taylor examined the relationships between a library system and its users, and categorized levels of user needs as follows [15]:

Q_1**:** the actual, but unexpressed need for information (the *visceral* need);

Q_2**:** the conscious, within-brain description of the need (the *conscious* need);

Q_3**:** the formal statement of the need (the *formalized* need);

Q_4**:** the question as presented to the information system (the *compromised* need).

Taylor argued that we always start from Q_1, in which our knowledge structure is very unstable. As we acquire background information through interactions with our colleagues and librarians, our knowledge structure gradually becomes stable, and we can proceed to Q_2, Q_3 and Q_4 step by step. Suchman also argued that all human activities including information retrieval are ad hoc *situated* actions, in which we cannot determine plans to reach our goals (answers) beforehand [16].

In summary, Taylor and Suchman insisted that our knowledge for finding answers is initially insufficient. This means that factors (ii) and (iii) are essential in information retrieval. In addition, we often ask incomplete questions, because too much effort is required to frame complete questions, so factor (iv) is also necessary. To deal with these factors, not only one response from a system, but also some kind of interaction with the users is required. Sadek stated that the following interactions are required: negotiation ability, contextual interpretation and language flexibility, interaction flexibility, and cooperative reactions [17].

1.2.3 Previous Studies and the Author's Approach

As the author previously discussed, some sort of interaction with users is required to resolve the various gaps between user questions and answers. In addition, the author pointed out that expression gaps beyond the keyword level still cannot be adequately resolved. This subsection reviews previous work on resolving the various gaps, and the author's approach in coping with the problems they encountered.

Although basic keyword-based text retrieval systems have no capabilities for interactions, some studies have tried to implement the capabilities to systems. These studies have mainly employed texts, keywords, and clusters as methods for ask-backs. Methods using texts as ask-backs include relevance feedback, which was examined within SMART system [18], and those using keywords include the system showing keywords related to user queries. These methods are also employed by contemporary web search engines such as Google and Excite. Cluster-based methods are employed by some systems such as Scatter/Gather [19] and WEBSOM [20], where each cluster is expressed as representative texts or keywords that are in the cluster. However, all the above have limitations, because they use either keywords or texts for ask-backs; keywords are too abstract, and texts are too specific.

In contrast, studies on resolving the gaps based on artificial intelligence techniques began in the 1980s, and some methods have been implemented as expert systems, including the Unix Consultant (UC) [21]. UC has the capability for ask-backs with user questions using natural languages if the questions are ambiguous. However, such classical expert systems depend on formal languages to represent knowledge, which require heavy construction and maintenance costs and make scaling up quite difficult.

Studies on text retrieval systems and QA systems to resolve expression gaps have only taken notice of gaps at the keyword level. Studies on text retrieval systems have used query expansion using term clustering [22], thesauri [23], latent semantic indexing [24], and other techniques. Recent studies on QA systems have also used thesauri or something approximating them; Harabagiu [25] used WordNet, and Kurohashi *et al.* [26] used a domain ontology. The expression gaps beyond keyword level cannot be adequately resolved by these methods, because more sophisticated natural language processing, including analysis of phrases and interpretations of metonymy, is required.

Now let us summarize the above discussion. As the author mentioned at the beginning of this chapter, there is a potential answer to any of our questions that exists somewhere in a large text collection; however, various gaps prevent us from accessing the appropriate texts to answer our questions. The author pointed out that interactions are necessary to resolve these gaps. Such interactions have mainly been studied through two approaches: extensions to keyword-based text retrieval systems, and the application of artificial intelligence techniques. However, these approaches have had limitations: the former, *i.e.,* ask-backs through keywords or texts, has been insufficient to resolve the gaps; while the

latter has not been able to be applied to existing large text collections. In addition, expression gaps beyond the keyword level had still not been adequately resolved in the previous studies.

This thesis proposes a solution to dealing with the various gaps, by advancing an approach employed in open-domain QA systems; that is, by utilizing NLP techniques such as full-parsing, various expression gaps at and beyond the keyword level can be resolved, and a user can be navigated from a vague question to specific answers that meet his/her situation through interactions. This approach has definite advantages over the others: unlike extensions to keyword-based text retrieval systems, this approach enables ask-backs, using phrases, not keywords or texts, and in contrast to the other methods based on artificial intelligence techniques, this can be applied to real-world large text collections.

Note that the resolution of some gaps has had to be left as future issues that need to be addressed; it is still difficult for the current NLP techniques to resolve some expression gaps beyond the keyword level, including anaphora, ellipses, and metaphors in dialogs. The resolution of belief gaps will require user modeling, but none of the present studies on user modeling can be applied to large text collections.

1.3 Overview of Dialog Navigator

Based on the real-world text collections provided by Microsoft Corporation, the author implemented the proposed methods on a system, the *Dialog Navigator*, which targeted common users of Windows PCs. The system started service on April 2002 at Microsoft Corporation's website.

Figure 1.1 shows the user interface for the Dialog Navigator. It has two frames: the upper shows dialog histories between the user and system, with an input box for the next user utterance; while the lower offers various choices. The user accesses the system with a web browser, and inputs a question in the input box with the keyboard. Next, the system does ask-back, by showing some choices in the bottom frame. If the user selects his/her choice with a mouse, the system displays an answer text, or does other ask-backs.

Now let us look at some dialogs example between users and the Dialog Navigator. This one clarifies a vague question through dialog:

(1.9) U1: エラーになった 'An error occurred'

Figure 1.1: User interface for *Dialog Navigator*

S1: エラーはいつ発生しますか? 'When did the error occur?'
1. Windows 起動中 'While booting Windows'
2. 印刷時 'While printing out'
3. ...

U2: *Windows* 起動中 'While booting Windows'

S2: あなたがお使いの *Windows* を選んでください. 'Which version of Windows are you using?'
1. *Windows 95*
2. *Windows 98*
3. ...

U3: *Windows 98*

S3: 以下の選択肢から選んでください. 'Please select from the following choices.'
1. W98:起動時のフォントドライバが読み込めないエラー 'W98: An error in trying to load font drivers on start-up'
「JIS フォントドライバーがインストールされていません」等のフォントの読み込みに関するエラーが発生した 'Font loading errors such as "JIS font driver has not been installed" were caused'
2. W98:起動時に「＜ファイル名＞が不正かありません」のエラーについて 'W98: The error "[filename] is illegal or not found" occurred on start-up'
3. ...

The system did two ask-backs with the user, clarifying his/her vague question. In addition,

when the system displayed answer texts as choices, it did not show these texts per se, but showed kinds of summaries as specific situations.

The following two dialogs are examples where user questions are precisely and flexibly matched with texts:

(1.10) U1: 行を追加したい 'I want to add columns.'

S1: 以下の選択肢から選んでください. 'Please select from the following choices.'
1. スプレッドシートで行や列を挿入、削除する'Insert or delete columns or rows on spreadsheets'
2. Excel で空白のセル、行、列を挿入する'Insert empty cells, columns, or rows in Excel'
3. PowerPoint で表に行、列を追加する 'Insert columns or rows on tables in PowerPoint'
4. グラフで空白のセル、行、列を挿入する'Insert empty cells, columns, or rows in Graph'
5. ...

(1.11) U1: 行が追加できない 'I can't add columns.'

S1: 以下の選択肢から選んでください. 'Please select from the following choices.'
1. Excel: 行の挿入 'Excel: Inserting columns'
2. XL2002: 共有ファイルで行が自動挿入される'XL2002: Columns are automatically (involuntarily) inserted on a shared file'
3. XL2000: コピーした行を挿入すると Excel がクラッシュする 'XL2000: Excel will crash if copied columns are inserted'
4. SQL: 一意な非クラスタ化インデックス付きテーブルに重複行が挿入 'SQL: Duplicated columns are inserted on unique non-clustered indexed tables'
5. ...

In these dialogs, a few relevant texts were shown to the user to answer his/her questions: the two synonymous phrases ("行を追加" (*gyō-wo tsuika*, add columns) and "行を挿入" (*gyō-wo sōnyū*, insert columns)) matched, and coordinations in text sentences were properly handled. In addition, text collections were selected depending on question types: in (1.10), on-line help texts were displayed because the user asked how he/she could add columns, and in (1.11), trouble shooting texts were displayed because the user identified the symptom.

1.4 Outline of Thesis

As the author pointed out at the beginning, vagueness gaps between user questions and texts create big problems in many situations around our world. Dialogs that clarify such vagueness are necessary for the real-world system that will replace call centers and experts.

To achieve such dialogs using real-world text collections, the thesis proposes a system based on the following methods:

- *Precise and flexible matching methods.*

  To achieve dialogs that will clarify vague questions, it is necessary to precisely and flexibly match the user question with the texts.

  Chapter 2 described several NLP-oriented methods that would bring such matching into being:

  - analyzing syntactic structures using a fairly accurate Japanese parser KNP [27],
  - resolving expression gaps using a dictionary, that contains synonyms, synonymous phrases, hypernyms, and hyponyms,
  - selecting text collections based on estimates of question types and detection of product names, and
  - matching score calculations based on the dependency structures of both user questions and text sentences.

  Chapter 3 extended these methods, *i.e.,* matching of metonymic expressions. Metonymy is a figure of speech in which the name of one thing is substituted for that of something else to which it is related [28]: *e.g.,* in the sentence "*GIF* を表示する" (*GIF-wo hyōji-suru*, display a GIF), "*GIF*" virtually indicates "*GIF* の画像" (*GIF-no gazō*, GIF image). Handling metonymic expressions is very important, because such expressions frequently occur both in user question and texts, and often raise gaps in syntactic structures. The extended method consists of:

  - automatic acquisition of metonymic expressions and their interpretative expressions from large corpora, including numerous user questions collected by Dialog Navigator, and
  - resolution of syntactic structure gaps using the acquired expressions.

  Both Chapter 2 and Chapter 3 had evaluations with testsets, and proved the effectiveness of the methods.

- *A strategy for efficiently clarifying vague questions.*

It is necessary to do ask-backs with the user to clarify his/her vague questions. For efficient ask-backs, the author proposes a strategy of complementarily using two methods.

Chapter 4 described those two methods, that is to say:

- *Description extraction:* a bottom-up approach.

  In most cases, the neighborhoods of the part that matches the user question describe specific symptoms and conditions of the problem that he/she often encounters. The system automatically extracts such neighborhoods (called *descriptions* here) based on syntactic structures.

- *Dialog cards:* a top-down approach.

  If a user's question is too vague, the bottom-up approach often fails, because it is difficult to detect parts matching the user's question. To deal with the problem, the system uses the know-how of Microsoft Corporation's call center. Namely, the system has *dialog cards* through which it does ask-backs for frequently asked questions that are vague.

- *Architecture for Real-world Operation.*

  To integrate these methods into a real-world system, several necessities, such as an interface for efficient dialogs, robust dialog controls, and user friendly outputs, are required.

  Chapter 5 discussed a total architecture and real-world evaluation of the system, that is, the *Dialog Navigator*. It has been accessible to the public on Microsoft Corporation's website from April 2002. The proposed methods were evaluated by analyzing the question logs for Dialog Navigator, in terms of the following three aspects:

  - Does the system return relevant answers?
  - How do users and the system behave in each dialog?
  - How do the results of description extraction help users?

  The results for each evaluation proved the effectiveness of the proposed methods.

Chapter 6 concluded this thesis.

Chapter 2

Precise and Flexible Matching Methods of User Questions with Texts

2.1 Introduction

"Information retrieval" in a broad sense is to seek information that satisfies user's demand in the real world. However, almost all previous studies on information retrieval have targeted a problem that a system matches a user question (*i.e.* sentences or a collection of keywords) with a lot of texts. That is to say, these studies were based on an assumption that information queries are described as sentences or collections of keywords, and that information in the real world is described as texts.

After 1990s, the amount of available texts has rapidly increased as the Internet became popular, and information retrieval techniques are required more and more. Applications of the techniques include not only conventional text retrieval systems, but also question answering systems, dialog systems, and example based machine translation systems.

As we have mentioned in Chapter 1, until 1980s, most of studies on information retrieval approximated both user questions and texts to collections of keywords. NLP-oriented approaches, like full parsing and semantic analysis, were also studied, but no significant improvement was achieved. Salton [10] evaluated several methods on SMART system, and reported that a simple indexing process based on the assignment of weighted terms to texts and search requests produced better retrieval results than a more sophisticated content analysis based on syntactic analysis.

In 1990s, along with the improvement of NLP, the applications of NLP techniques for information retrieval revived.

First, several studies for applying NLP techniques on text retrieval were done in Text REtrieval Conference[1] (TREC). In TREC-2, Strzalkowski *et al.* [29] tried out weighting on using four types of head-modifier pairs given by a grammar-based parser, and reported that such weighting improved performance. And then, NLP Track, which provided a more focused look at how NLP techniques can help in achieving better performance in text retrieval, were organized in TREC-5. In this track, GENLP system [30] and CLARIT system [31] applied full-parsing-based methods (*i.e.* using head-modifier pairs), and reported some improvements of performance. But, that improvements were limited as compared with costs for deeper syntactic analysis, and fewer studies for applying NLP techniques have been done in subsequent TREC tracks. Strzalkowski *et al.* [32] summarized TREC-5 NLP track as follows:

> ... natural language processing techniques have solid but limited impact on the quality of text retrieval, particularly precision.

Contemporary web search engines are also in line with the conventional text retrieval techniques: they do not depend on NLP techniques, excepting morphological analysis. The improvements of retrieval performance have mainly achieved by proximity matching techniques, and exploitation of HTML structures or hyperlinks.

Meanwhile, studies on text-based QA systems, which started in the former half of 1990s, have focused on applications of NLP techniques in general: MURAX [11], which is a QA system based on a encyclopedia, applied predicate-argument matches for answer re-ranking; FAQ Finder [8] applied case analysis and question type analysis for matching a user input with "questions" in FAQ files of USENET; and Dialog Help system of CIMS Kyoto University [26] also analyzed sentence structures for flexible matching of user query with natural language knowledge base, and realized basic dialog facilities such as asking-backs and contextual interpretations. This trend was taken over by recent studies on open domain QA systems of TREC QA tracks [5] and NTCIR QAC [9]. In these tracks, most of excellent systems adopted sentence structure analysis and question type analysis for finding relevant answers precisely [12–14].

Why are the NLP techniques effective not for text retrieval systems, but QA systems? It comes from the difference of their goals. The target of text retrieval is to find a text collection that satisfies user's query, and in most cases it is achieved by approximating

[1] `http://trec.nist.gov/`

each text as a collection of keywords. Needless to say, some queries require consideration of relations between keywords. However, the requirement is generally satisfied by proximity matching techniques, and the NLP techniques have little part to play. In contrast, the target of QA is to find "exact" answers from each text: proximity matches is insufficient, and it requires sentence structure analysis.

Also, the main target of this thesis, *i.e.,* the resolution of the gaps between user question and texts, requires the sentence structure analysis. First, to resolve expression gaps on a phrase level, *e.g.* between (2.1) and (2.2), a matching method based on sentence structures is required. Naive methods by which "読む" (*yomu*, read) and "受信する" (*jushin-suru*, receive) are regarded as synonyms do not work well, because these two keywords would have different senses in other contexts.

(2.1) メールを 読む
mēru-wo *yomu*
mail-acc read
'read mail'

(2.2) メールを 受信する
mēru-wo *jushin-suru*
mail-acc receive
'receive mail'

Secondly, to resolve vagueness gaps, *i.e.,* to make asking-backs for vague questions, the sentence structure analysis is also required: in many cases, the neighborhoods of the parts that match the question in texts are suitable for such asking-backs, because they make the difference between matched texts more clear (We describe the method for making such asking-backs in Chapter 4).

This chapter proposes several methods for achieving precise and flexible matching that satisfies the above purposes, based on full-parsing results of user questions and texts. First, Section 2.2 describes text collections that we target in this thesis. Next, Section 2.3 gives basic sentence structure analysis methods, including parsing and question type estimation. Section 2.4 gives two methods for resolving expression gaps between user questions and texts: *synonymous expression dictionary* and *ontological dictionary*. Section 2.5 shows indexing methods for retrieving relevant text fast. Section 2.6 shows two selection methods for finding exact texts: selection by question types and product names. Section 2.7 describes score calculation methods for improving precision, giving large points to matches

Table 2.1: Text collections

text collection	# of texts	# of characters	matching target
Glossary	4,707	700,000	entries
Help texts	11,306	6,000,000	titles
Support KB	23,323	22,000,000	entire texts

of modifier-head relations of Japanese sentences. And then, Section 2.8 evaluates the proposed methods on testsets, Section 2.9 compares our methods with those of previous work. Finally, Section 2.10 concludes this chapter.

2.2 Text Collections

In this thesis, we target three types of text collections provided by Microsoft Corporation as the knowledge base. Table 2.1 shows the text collections and their scales.

Glossary (Figure 2.1) has definitions of terms related to computers. Help texts (Figure 2.2) have instructions for Windows, Office, and other Microsoft's products. Support KB (Knowledge Base, Figure 2.3) is a large text collection provided by Microsoft Corporation. Mainly this collection consists of frequently asked questions for call centers, technical information of products, and trouble shooting information.

We limit matching targets of each text collection as follows:

Glossary each entry word is targeted for matching with user questions, because it corresponds with *"What is"* questions (*e.g.* "What is .NET?").

Help Texts the title of each help text is targeted for matching with user questions, because it corresponds with *"How to"* questions (*e.g.* How to send an e-mail by Outlook?)

Support KB entire texts are targeted for matching with user questions, because situations or symptoms of each problem are usually explained by the whole sentences of each text.

C#
.NET Framework 上のプログラミングのために最適化されたオブジェクト指向のプログラミング言語。'A programming language optimized for programming on the .NET framework.'
Microsoft .NET
2000 年にマイクロソフトが発表した、インターネットを包括的にサポートするソフトウェアの動作環境と開発環境の両方を提供する基盤技術。'The architecture that provides both operating and development environments of softwares, comprehensively supporting the Internet, released by Microsoft in 2000.'
クラスタリング 'clustering'
複数のコンピュータを統合し、あたかも 1 台の大規模なコンピュータ システムとして運用するための技術。'A technology for operating multiple computers by integrating them, as if they were one large-scale computer system.'
シフト JIS 'Shift-JIS'
パソコンで主に利用されている日本語文字の文字コード体系。'One of the Japanese character encoding schemes mainly used for personal computers.'

Figure 2.1: Glossary

2.3 Sentence Structure Analysis

This section proposes the method for getting sentence structures which are suitable for the matching, based on modifier-head (M-H) relation structures of Japanese sentences. This method is applied for both user questions and sentences in texts (We call them *text sentences* in the following sections).

2.3.1 Parsing and Keyword Extraction

First of all, both user questions and texts are parsed, using a morphological analyzer JUMAN [33] and a parser KNP [27] for Japanese. JUMAN segments sentences into words (words in Japanese sentences are usually not separated by white spaces) and detects their PsOS. KNP detects dependency relations between *bunsetsu* units. *Bunsetsu* is a commonly used linguistic unit in Japanese traditional grammar, consisting of one or more adjoining content words and zero or more following functional words [2].

After that, keywords are extracted from each *bunsetsu*. Words that are detected as the following PsOS are extracted:

[2]The English equivalent for *bunsetsu* would be a small noun phrase, a prepositional phrase, or a verb phrase consisting of auxiliary verbs and a main verb, and so on.

Excel で空白のセル、行、列を挿入する 'Inserting empty cells, columns, or rows in Excel'

1. 次のいずれかの操作を行います。 'Carry out one of the following operations:'

   空白セルを挿入する 'Inserting empty cells' 空白セルを挿入するセル範囲を選択します。'Highlight cells where to insert new cells.' 挿入するセルと同じ数のセルを選択します。 'The number of the highlighted cells should be same as that of inserting cells.'

   行を 1 行挿入する 'Inserting a column' 行を挿入する位置のすぐ下の行のセルを 1 つクリックします。'Click on a cell on the column just below the location to insert a column.' たとえば 5 行目の上に行を挿入する場合は、5 行目のセルをクリックします。'For example, if you want to insert a column above the fifth column, click on a cell on the fifth column.'

   複数の行を挿入する 'Inserting multiple columns' 行を挿入する位置のすぐ下の行を選択します。'Highlight the columns just below the location to insert columns.' 挿入する行数と同じ行数のセルを選択します。'The number of the highlighted columns should be same as that of inserting columns.'

   列を 1 列挿入する 'Inserting a row' 列を挿入する位置のすぐ右の列のセルを 1 つクリックします。 'Click on a cell on the just right row of the location to insert a row.' たとえば B 列の左に新しい列を挿入する場合は、B 列のセルをクリックします。 'For example, if you want to insert a row on the left of the row B, click on a cell on the row B.'

   複数の列を挿入する 'Inserting multiple rows' 列を挿入する位置のすぐ右の列を選択します。 'Highlight the rows on the just right row of the location to insert rows.' 挿入する列数と同じ列数のセルを選択します。 'The number of the highlighted rows should be same as that of inserting rows.'

2. [挿入] メニューの [セル]、[行]、[列] のいずれかをクリックします。'Click on either of [cell], [column], or [row] in [insert] menu.'

3. 行や列ではなく、セル範囲を移動またはコピーする場合、[挿入貼り付け] ダイアログボックスで、周囲のセルをシフトさせる方向をクリックします。 'If you want to move or copy not columns or rows but a cell area, click on the direction to shift the surrounding cells in [insert/paste] dialog box.'

Figure 2.2: A Help Text

Recycle Bin Settings Not Retained During Windows 2000 Upgrade (Q240433)

The information in this article applies to:

- Microsoft Windows 2000, Advanced Server
- Microsoft Windows 2000, Professional
- Microsoft Windows 2000, Server

SYMPTOMS
After you upgrade to Windows 2000, from Microsoft Windows NT 4.0, Microsoft Windows 95, or Microsoft Windows 98, your settings for the Recycle Bin may be different than they were before you upgraded.

CAUSE
This issue can occur because the settings for the Recycle Bin are not migrated during the upgrade.

RESOLUTION
To resolve this issue, right-click **Recycle Bin**, click **Properties**, configure the settings you want, and then click **OK**.

MORE INFORMATION
The Recycle Bin settings for Windows 95 and Windows 98 are in the registry. The Recycle Bin settings for Windows NT 4.0 are kept on disk.

Figure 2.3: Microsoft Support KB (an English version)

- common nouns.
- *sahen* nouns.
- proper nouns, including person names, place names, and organization names.
- numerals.
- verbs.
- adjectives.
- adjective verbs.
- adverbs.
- *katakana* words.
- Roman alphabet words.

Conjugations of verbs, adjectives, adjective verbs, adverbs are normalized into original forms.

Excepting that extremely general verbs are not extracted: “する” (*suru*, do), “行う”/“おこなう” (*okonau*, do), “ある” (*aru*, be), “行く”/“いく” (*iku*, go), “出来る”/“できる” (*dekiru*, can do), “下さる”/“くださる” (*kudasaru*; honorific) and “ございます” (*gozaimasu*; honorific).

2.3.2 Unification of *Bunsetsu*

It is a difficult issue to determine a base unit for the matching of user question with texts. In most cases the *bunsetsu* unit annotated by KNP is appropriate, there are some exceptions. Consider the following two sentences.

(2.3)

画面を	コピーできない
gamen-wo	*copī-deki-nai*
screen-acc	copy-(can)-(not)

‘(I) can not copy the screen’

(2.4)

画面コピーを	する	ことが	できない
gamen-copī-wo	*suru*	*koto-ga*	*deki-nai*
screen-copy-acc	(do)	(matter)-nom	(can)-(not)

‘(I) can not do screen copy’

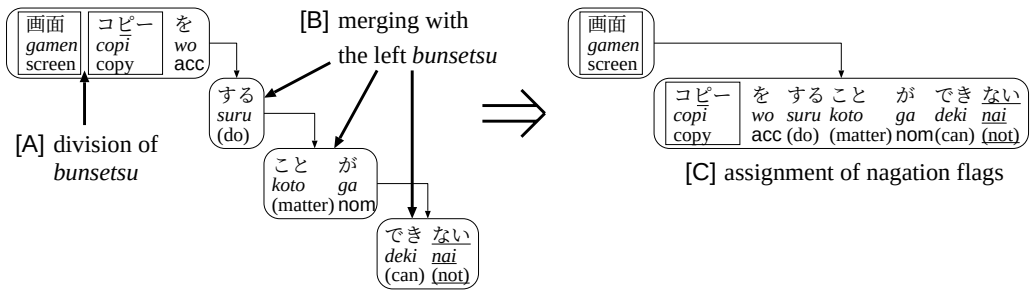


Figure 2.4: Unification of *bunsetsu* and assignment of negation flags

(“|” shows a boundary of *bunsetsu* annotated by KNP.)

These two sentences have almost the same meaning, but the analysis results of KNP are quite different: (2.3) is divided into two *bunsetsu*, and (2.4) is divided into four *bunsetsu*.

To match such sentences correctly, *bunsetsu* are divided and unified by the following rules (Figure 2.4):

1. If a *bunsetsu* has multiple keywords, it is divided into each correspondent single keyword. For example, a *bunsetsu* “画面 (*gamen*, screen) コピー (*copi*, copy) を (*wo*; acc)” is divided into two *bunsetsu*: “画面 (*gamen*, screen)” and “コピー (*copi*, copy) を (*wo*; acc)” (Figure 2.4 [A]). Note that it is supposed that the divided two adjacent *bunsetsu* have an M-H relation, and that the following functional word “を (*wo*; acc)” belongs to the latter *bunsetsu*.

2. If a *bunsetsu* forms *fukugō-ji* (compositive ancillary words of Japanese) or has no keywords, it is merged with the left *bunsetsu*. For example, “する (*suru*, do)”, “こと (*koto*, matter) が (*ga*; nom)” and “でき (*deki*, can) ない (*nai*, not)” are merged, because they have no keywords (Figure 2.4 [B]).

2.3.3 Assignment of Negation Flags

There are several variations of negative expressions in Japanese. For example, “ない” (*nai*; adjective), “ぬ” (*nu*; auxiliary verb), “非” (*hi*; prefix), or “不” (*fu*; prefix) mean negation. To resolve gaps of these negative expressions between user questions and texts, *negation flags* are assigned to *bunsetsu* that contain negative expressions (Figure 2.4 [C]).

Table 2.2: Question types

question type	percentage	description	question-pattern rules *(e.g.)*
what type	$\simeq 10\%$	The user asks some fact.	～って何ですか 'What is ...?'; ～の説明をして 'Make an explanation about ...'; ～の意味を教えて 'Tell me what ... mean'
how type	$\simeq 35\%$	The user asks how to do something.	～方法を教えて 'Tell me how to ...'; ～にはどうしたらいいの 'What am I going to do for ...?'; ～の使い方 'How to ...?'
symptom type	$\simeq 50\%$	The user shows some symptom since he/she want to know how to cope with it.	～してしまう 'end up ...'; ～が使えません 'I cannot use ...'; ～ができない 'I cannot do ...'
no type	$\simeq 5\%$	The user asks other types of questions, including out-of-domain questions.	あなたの名前は? 'What is your name?'; こんにちは 'Hello'; ～が欲しい 'I want ...'

2.3.4 Question Type Estimation

In order to select text collections (Subsection 2.6.1) appropriately, question types of user questions are estimated using question-pattern rules.

Table 2.2 shows the four question types that our method employs. These types were derived by our preliminary examination of the question logs for the natural language based text retrieval system *Hanashi-kotoba Kensaku*[3] serviced by Microsoft Japan.

The estimation of question types is based on the longest matching of question-pattern rules from the end of user question. Table 2.2 right shows the examples of question-pattern rules. This method works well in most cases, because Japanese is head-final, and the final expression shows a question type [26].

[3]http://www.microsoft.com/japan/enable/nlsearch/

2.3.5 Removal of Final Expressions

To avoid ineffective matches with texts, some final expressions that match particular question-pattern rules are removed from user questions. Such final expressions are usually useless for the matching.

For example, if a user question ends in "〜って何ですか" (*...tte nan-desu-ka*, What is ...; *what* type) or "〜方法を教えて" (*... hōhō-wo oshiete*, Tell me how to ...; *how* type), these final expressions are removed.

2.4 Expression Gap Resolution

The expression gaps between user questions and texts are a big problem for the matching. To cope with this problem, two types of dictionaries are used: *synonymous expression dictionary* and *ontological dictionary*.

2.4.1 Synonymous Expression Dictionary

In addition to synonyms (keyword level), there are a great deal of synonymous phrases (phrasal level) such as "パソコンを起動する" (*pasokon-wo kidō-suru*, boot a PC), "Windowsを起動する" (*Windows-wo kidō-suru*, boot Windows), and "電源を入れる" (*dengen-wo ireru*, switch on). Our method resolves such phrasal gaps using the *synonymous expression dictionary*.

Figure 2.5 shows a part of synonymous expression dictionary. It groups keywords and phrases as synonymous expressions (We call these groups as *synonymous expression groups*). The groups of synonymous keywords were mainly derived from a synonym dictionary constructed by Microsoft Corporation. In contrast, the groups of synonymous phrases were made by analyzing the question logs for *Hanashi-kotoba Kensaku*: to be exact, we extracted frequently occurred phrases, and then manually grouped them.

Our method utilizes this dictionary as follows:

1. Each expression in the dictionary is analyzed by the method described in Section 2.3.

2. Recursive relations in the dictionary are automatically expanded. Figure 2.6 shows an example of such relations. Consider the following expression.

[**発生** *hassei*]
発生, 起きる, 起こる
hassei *okiru* *okoru*
'occur' 'occur' 'occur'
[**使う** *tsukau*]
使う, 使用, 使える
tsukau *shiyō* *tsukaeru*
'use' 'use' 'can use'
[**読む** *yomu*]
読む, 読み込む
yomu *yomikomu*
'read' 'read .. into'
[**メール** *mēru*]
メール, メイル, *e-mail*
mēru *meiru* *e-mail*
'mail' 'mail' 'e-mail'
[**メールを読む** *mēru-wo yomu*]
メールを読む, メールを受信する, メッセージを読む, メッセージを受信する
mēru-wo yomu *mēru-wo jushin-suru* *messēge-wo yomu* *messēge-wo jushin-suru*
'read a mail' 'receive a mail' 'read a message' 'receive a message'
[**パソコンを起動する** *pasokon-wo kidō-suru*]
パソコンを起動する, *Windows*を起動する, 電源を入れる
pasokon-wo kidō-suru *Windows-wo kidō-suru* *dengen-wo ireru*
'boot a PC' 'boot Windows' 'switch on'

Figure 2.5: Synonymous expression dictionary

(2.5) メールを 読む
mēru-wo *yomu*
mail-acc read
'read a mail'

It contains two keywords: "メール" (*mēru*, mail) and "読む" (*yomu*, read). The former has two synonyms: "メイル" (*meiru*, mail) and *"e-mail"*. Also the latter has a synonym: "読み込む" (*yomikomu*, read ... into). In this case, that phrase is preliminarily expanded into $3 \times 2 = 6$ phrases.

Note that each structure of expanded phrase is preserved.

3. Synonymous expression groups are extracted from each analyzed result of a target sentence (both a user question and text sentence), by looking up the expanded synonymous expression dictionary. This extraction is based on the structures of both target sentences and phrases of the dictionary. If a part of a target sentence is perfectly matched with a phrase of a synonymous expression group (including keywords in each *bunsetsu* and M-H relations between *bunsetsu*), the group is extracted.

   In Figure 2.7, four synonymous expression groups [**使う** *tsukau*], [**メール** *mēru*], [**読む** *yomu*], and [**メールを読む** *mēru-wo yomu*] are extracted from a user question.

   To reduce response time for a user input, synonymous expression groups are preliminarily extracted from text sentences, and an index is created (see Section 2.5).

4. To match the groups extracted from both a user question and a text sentence at the score calculation process (see Subsection 2.7.1), the information which *bunsetsu* and M-H relations each group is extracted from is kept. For example, in Figure 2.7, the following information is kept: a group [**メールを読む** *mēru-wo yomu*] is extracted from *bunsetsu* **C, D** and the M-H relation between **C** and **D**.

2.4.2 Ontological Dictionary

In some cases, the expression gaps between user questions and texts can not be properly resolved by the synonymous expression dictionary. Consider the following two expression gaps: "ブラウザ" (*brauza*, browser) $\Longleftrightarrow$ *"IE5"* (Internet Explorer 5) and "ブラウザ" (*brauza*, browser) $\Longleftrightarrow$ *"IE6"* (Internet Explorer 6). If the three keywords ("ブラウザ"

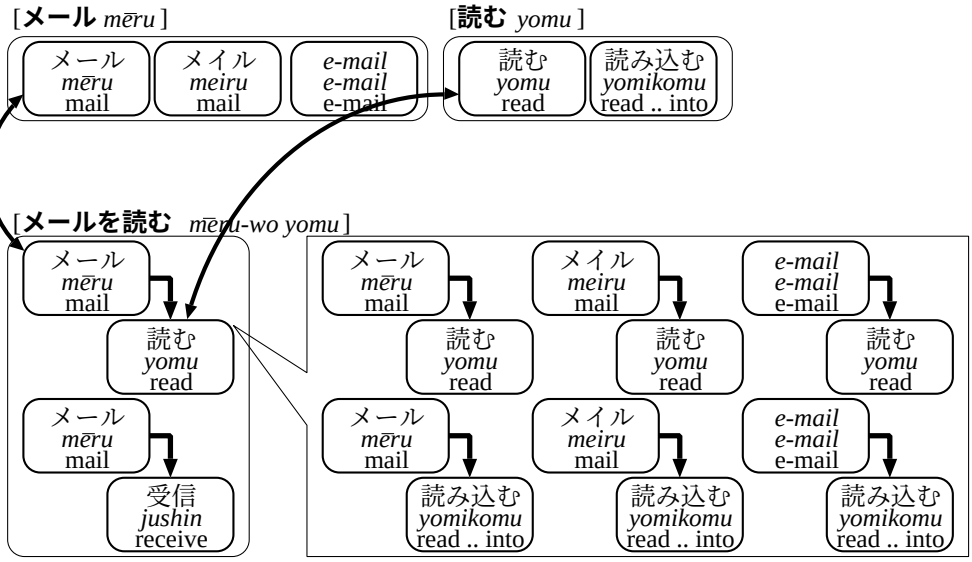


Figure 2.6: Expansion of recursive relations

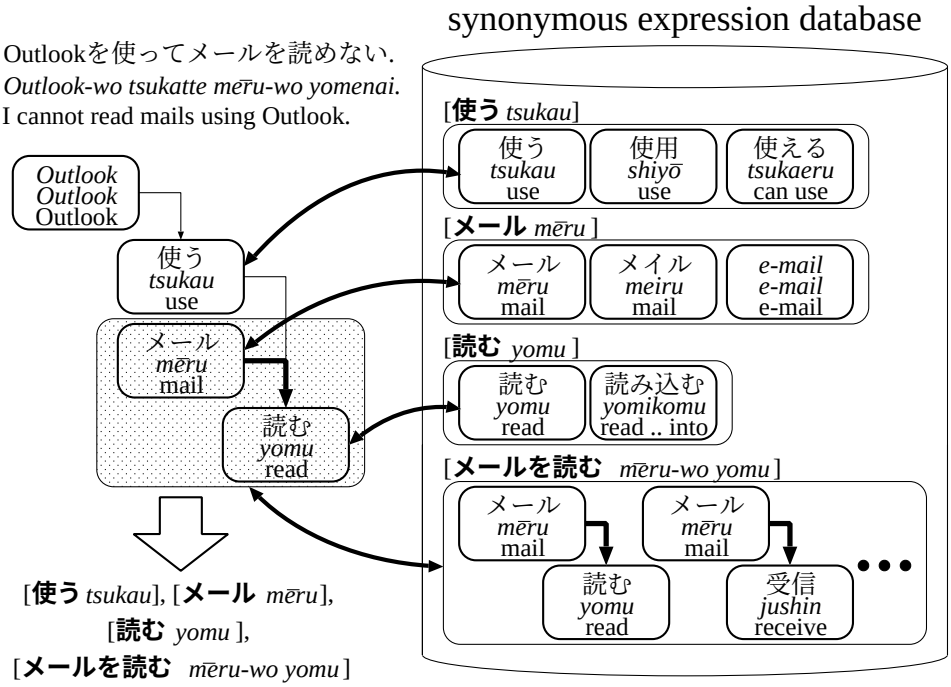


Figure 2.7: Extraction of synonymous expression groups from a user question

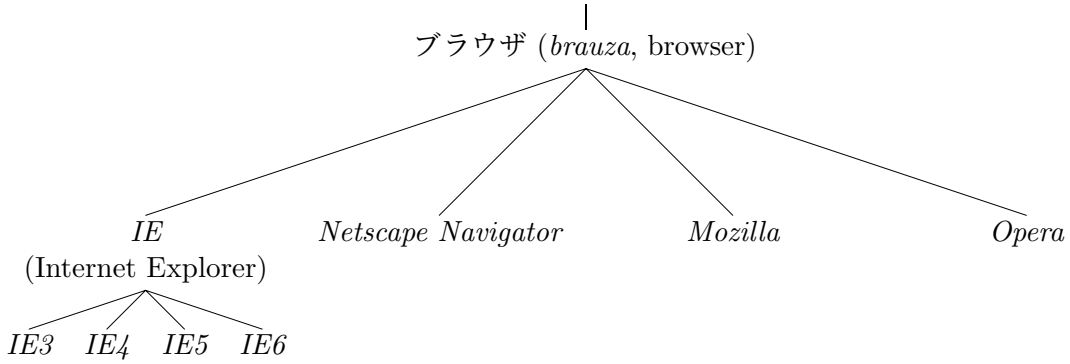


Figure 2.8: Ontological dictionary

(*brauza*, browser), *"IE5"*, and *"IE6"*) are regarded as synonyms, a problem occurs. For a question about "*IE6*", texts about "*IE5*" are incorrectly shown.

To cope with this problem, our method uses an *ontological dictionary* as shown in Figure 2.8, and expands hypernyms and hyponyms of keywords in texts. If any keyword of this dictionary occurs in a text, its hypernyms and hyponyms are also regarded as keywords of the text. For example, if *"IE6"* occurs in a text, its hypernyms (*"IE"* and *"ブラウザ"* (*brauza*, browser)) are also dealt as keywords. If *"IE"* occurs in a text, its hypernyms ("ブラウザ" (*brauza*, browser)) and its hyponyms (*"IE3", "IE4", "IE5",* and *"IE6"*) are also regarded as keywords. Our method avoids the above problem by suppressing such expansions for user questions.

2.5 Indexing

To retrieve texts fast, our method creates an index of keywords and synonymous expression groups in advance.

Our method extracts keywords and entries of synonymous expressions from the user question, and then looks up them in the inverted index. It selects texts which contain at least one keyword or synonymous expression. If the number of selected texts exceeds 1000, the best 1000 texts that have more matched keywords or synonymous expressions are selected.

Table 2.3: Selection of text collections by question types

text collection		question type *what*	*how*	*symptom*	*no*
Glossary	*what* type	o			o
Help texts	*how* type	o	o		o
Support KB	*symptom* type	o		o	o
	how type	o	o		o
	no type	o	o	o	o

2.6 Selection of Text Collections

To match a user question with texts more exactly, our method selects text collections by question types and product names.

2.6.1 Selection by Question Types

As we have mentioned in Subsection 2.3.4, our method extracts question types (*symptom*, *how*, or *what*) of the user input. The text collections can be also classified into those three types. It seems that Glossary corresponds to *what* type, Help texts to *how* type, and Support KB to *how* or *symptom* type. Hence our method can precisely select texts by those types.

Text collections are selected as shown in Table 2.3, based on question type estimation described in Subsection 2.3.4. Basically, *what* type questions are answered by Glossary, and *how* type questions are answered by Help texts. About Support KB, our method uses tags that indicate *how* type or *symptom* type.

Note that our method selects all the text collections for *what* type questions, showing users Glossary texts first. It is because some *what* type questions may be interpreted as other types. For example, a *what* type question "コントロールパネルについて教えて" (*kontorōrupaneru-ni tsuite oshiete*, Could you tell me about Control Panel?) may be interpreted that a user asks not about the definition of Control Panel, but how to use Control Panel.

2.6.2 Selection by Product Names

Texts are generally classified under products (*e.g.* Windows, Word, and Excel) as shown in Figure 2.3. Our method also selects texts by product names occurred in the question.

If product names occur in the question, texts for these products are selected. If multiple product names occur (*e.g.* "*Excel*で作った表が *Word*で読み込めない" (*Excel-de tsukutta hyō-ga Word-de yomikomenai*, I can not read a table made by Excel into Word)), texts tagged with any of the product names are selected.

2.7 Score Calculation

Our method calculates the score of each text that is selected by looking up the index (see Section 2.5) and then selected by question types and product names (see Section 2.6). The calculation of each text score are based on similarity between the user question and a sentence in the text. To improve precision, large points are given to matches of M-H relations of Japanese sentences.

2.7.1 Sentence Similarity Calculation

Similarity between a user question and a text sentence is calculated as a product of both sentences' ratios of which *bunsetsu* units and head-modifier relations have correspondence with the other's *bunsetsu* units *(coverage)*, based on unified *bunsetsu* units described in Section 2.3.

First, our method makes *correspondence* of *bunsetsu* units and head-modifier relations between the two sentences, and gives *correspondence score* (c; $0 \leq c \leq 1$) for each of them, according to the following conditions:

1. If a keyword in $\mathbf{A}$ (a *bunsetsu* unit of a user question) corresponds with a keyword in $\mathbf{A'}$ (a *bunsetsu* unit of a text sentence), our method makes correspondence between $\mathbf{A}$ and $\mathbf{A'}$. The correspondence score $c_{\mathbf{A},\mathbf{A'}}$ is calculated as follows:

   (a) If $\mathbf{A}$ and $\mathbf{A'}$ have same keywords,

   $$c_{\mathbf{A},\mathbf{A'}} = \frac{(\#\text{ of keywords that are in both } \mathbf{A} \text{ and } \mathbf{A'})}{\max(\#\text{ of keywords in } \mathbf{A}, \#\text{ of keywords in } \mathbf{A'})}. \tag{2.6}$$

   For example, if $\mathbf{A}$ is *"Windows 98 SE"* (three keywords) and $\mathbf{A'}$ is *"Windows 98"* (two keywords), $c = 2/3 (\simeq 0.67)$, because two keywords *"Windows" "98"*

are in both $\mathbf{A}$ and $\mathbf{A}'$. Note that in most cases $c_{\mathbf{A},\mathbf{A}'} = 1.0$, because most *bunsetsu* has only one keyword.

(b) If a keyword in $\mathbf{A}$ matches a hypernym/hyponym of it in $\mathbf{A}'$, $c_{\mathbf{A},\mathbf{A}'} = 0.9$ (discounted score).

(c) If negation flags of $\mathbf{A}$ and $\mathbf{A}'$ do not match, $c_{\mathbf{A},\mathbf{A}'}$ is 0.6 times as big as in the case that the flags match.

2. For an M-H relation in a user question $\mathbf{A} \rightarrow \mathbf{B}$ and an M-H relation in a text sentence $\mathbf{A}' \rightarrow \mathbf{B}'$ ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{A}'$, and $\mathbf{B}'$ are *bunsetsu* units, and $\rightarrow$ shows that the left *bunsetsu* depends on the right *bunsetsu*), if $\mathbf{A}$ corresponds $\mathbf{A}'$, and $\mathbf{B}$ corresponds $\mathbf{B}'$, our method makes correspondence between $\mathbf{A} \rightarrow \mathbf{B}$ and $\mathbf{A}' \rightarrow \mathbf{B}'$. The correspondence score $c_{\mathbf{A}\rightarrow\mathbf{B},\mathbf{A}'\rightarrow\mathbf{B}'}$ is calculated as follows:

$$c_{\mathbf{A}\rightarrow\mathbf{B},\mathbf{A}'\rightarrow\mathbf{B}'} = c_{\mathbf{A},\mathbf{A}'} \cdot c_{\mathbf{B},\mathbf{B}'}. \tag{2.7}$$

3. If a synonymous expression group extracted from a user question is also extracted from a text sentence, our method makes correspondence between *bunsetsu* units and M-H relations from which the groups are extracted (Figure 2.9). The correspondence score $c = 1.0$.

As a result, every *bunsetsu* and M-H relation of both sentences has correspondence score. If it has no correspondence, its score equals zero. And if it has multiple correspondences, the maximum score is regarded as its score.

The *coverage* C of a sentence (both a user question and a text sentence) is calculated as follows:

$$C = \frac{\sum_{\mathbf{X}\in\mathbb{B}} c_{\mathbf{X}} + m \cdot \sum_{(\mathbf{X_1}\rightarrow\mathbf{X_2})\in\mathbb{R}} c_{\mathbf{X_1}\rightarrow\mathbf{X_2}}}{n(\mathbb{B}) + m \cdot n(\mathbb{R})} \tag{2.8}$$

where $\mathbb{B}$ is the collection of *bunsetsu*, $\mathbb{R}$ is the collection of M-H relations, $n()$ means the number of elements (*bunsetsu* or modifier-head relations) in a collection, and m is a parameter for weighting on matches of M-H relations ($m \geq 0$). Note that we can change m to improve precision of matching.

Finally, the *similarity* S between the user question and the text sentence is calculated as follows:

$$S = C_U \cdot C_T \tag{2.9}$$

where C_U is the coverage of the user question, and C_T is the coverage of the text sentence.

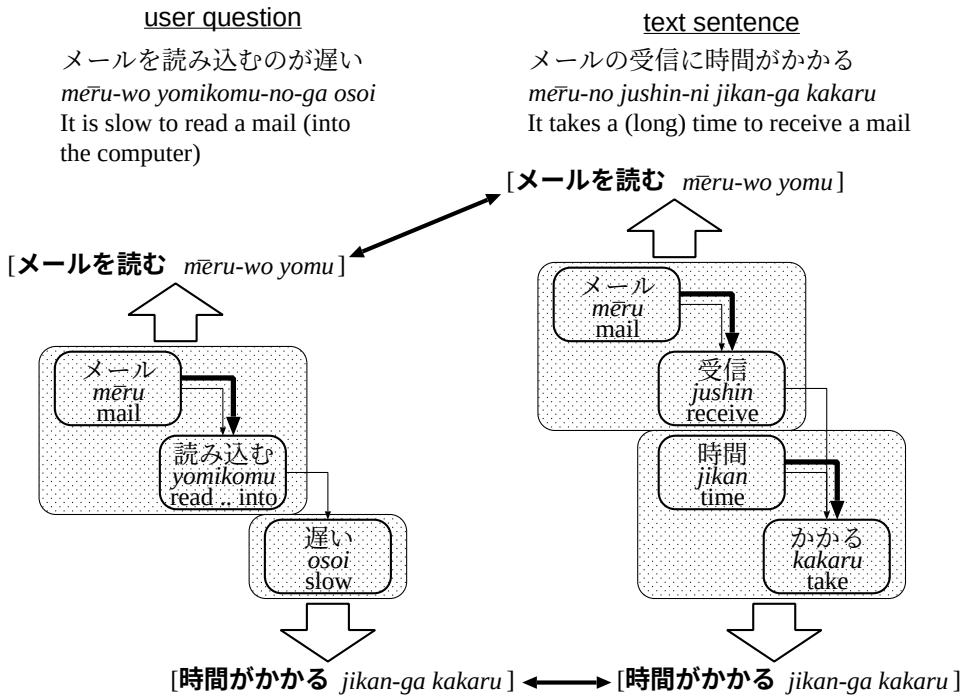


Figure 2.9: Making correspondences of synonymous expressions

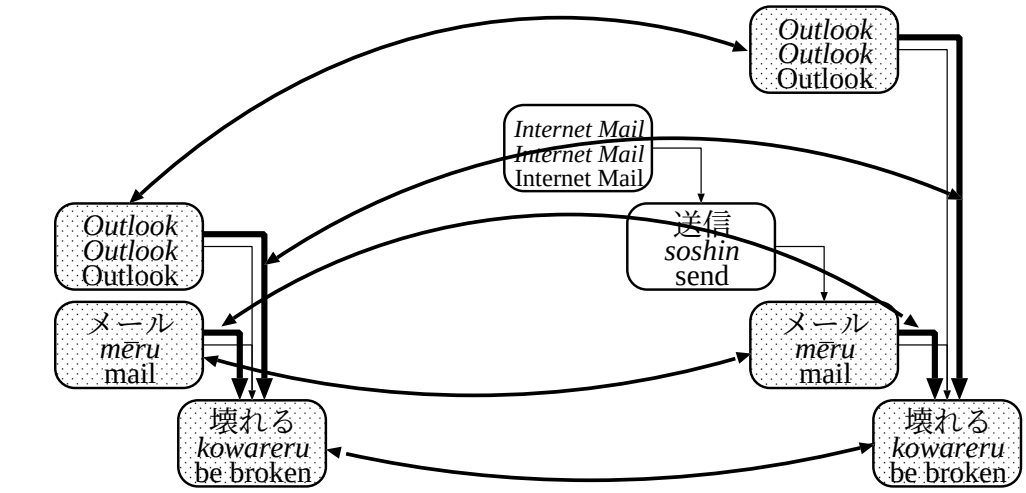
In Figure 2.10, both the user sentence and the text sentence has three correspondences of *bunsetsu* and two correspondences of M-H relations, and all the correspondence scores c are 1.0. As a result, the coverage is $C_U = 1.0$ and $C_T = 0.54$ respectively, and then the similarity $S = 0.54$.

2.7.2 Representative Sentences and Scores of Texts

Our method selects the sentence which has the largest similarity in a text as the *representative sentence* of the text. For Glossary and Help texts, these entries or titles are selected as the representative sentence, because each of those has only one sentence as the matching target. The similarity of the sentence is regarded as the score of the text.

2.7.3 Special Score Calculation for Support KB

Because Support KB has very large texts and the whole of them are the matching target as shown in Table 2.1, our method applies special calculating scores for Support KB as



3.0	—	$\sum_{\mathbf{X}\in\mathbb{B}} c_{\mathbf{X}}$: the sum of correspondences of *bunsetsu*	—	3.0
2.0	—	$\sum_{(\mathbf{X_1}\to\mathbf{X_2})\in\mathbb{R}} c_{\mathbf{X_1}\to\mathbf{X_2}}$: the sum of correspondences of M-H relations	—	2.0
3	—	$n(\mathbb{B})$: the number of *bunsetsu*	—	5
2	—	$n(\mathbb{R})$: the number of M-H relations	—	4
1.0	—	C: coverage ($m = 1.0$)	—	0.54

$$S = 1.0 \cdot 0.54 = 0.54$$

Figure 2.10: Making correspondences between a user question and a text sentence, and similarity calculation

follows:

$$S^i = p \cdot C_U^i = p \cdot \frac{\sum_{\mathbf{X} \in \mathbb{B}} c'^i_{\mathbf{X}} + m \cdot \sum_{(\mathbf{X_1} \to \mathbf{X_2}) \in \mathbb{R}} c'^i_{\mathbf{X_1} \to \mathbf{X_2}}}{n(\mathbb{B}) + m \cdot n(\mathbb{R})} \tag{2.10}$$

where S^i is the similarity between a user question U and a text sentence T_i; C_U^i is coverage of U with T_i; p is a factor for considering which part T_i is in; $c'^i_{\mathbf{X}}$ and $c'^i_{\mathbf{X_1} \to \mathbf{X_2}}$ are respectively correspondences of a *bunsetsu* $\mathbf{X}$ and an M-H relation $\mathbf{X_1} \to \mathbf{X_2}$ with those of T_i, which consider proximity matches with neighbor sentences. Each factor has the following meaning:

- Our method ignores C_T^i (coverage of a text sentence T_i), because lengths of text sentences are not uniform. Namely, S^i depends only on C_U^i.

- Our method considers proximity matches with neighbor sentences, because a phenomenon is often described by more than one sentence. That is, when the system calculates the similarity between U and T_i, it modifies correspondence scores ($c^i_{\mathbf{X}}$ and $c^i_{\mathbf{X_1} \to \mathbf{X_2}}$) with the before sentence T_{i-1} and the next sentence T_{i+1} as follows:

$$c'^i_{\mathbf{X}} = \max(\frac{1}{2}c^{i-1}_{\mathbf{X}}, c^i_{\mathbf{X}}, \frac{1}{2}c^{i+1}_{\mathbf{X}}) \tag{2.11}$$

$$c'^i_{\mathbf{X_1} \to \mathbf{X_2}} = \max(\frac{1}{2}c^{i-1}_{\mathbf{X_1} \to \mathbf{X_2}}, c^i_{\mathbf{X_1} \to \mathbf{X_2}}, \frac{1}{2}c^{i+1}_{\mathbf{X_1} \to \mathbf{X_2}}) \tag{2.12}$$

- As shown in Figure 2.3, each text of Support KB has structures: subjects; and sections such as "概要" (*gaiyō*, ABSTRACT), "症状" (*shōjō*, SYMPTOMS), "原因" (*gen-in*, CAUSE), "解決方法" (*kaiketsu-hōhō*, RESOLUTION) and "関連情報" (*kanren-jōhō*, MORE INFORMATION). In many cases, subjects and sections such as "概要" (*gaiyō*, ABSTRACT) and "症状" (*shōjō*, SYMPTOMS) corresponds with frequently asked questions that describe typical situations. Therefore, our method gives weighting p according to where T_i exists as follows:

  - subjects and "概要" (*gaiyō*, ABSTRACT) $p = 1.0$
  - "現象" (*genshō*, PHENOMENON) and "症状" (*shōjō*, SYMPTOMS) $p = 0.8$
  - other sections $p = 0.6$

2.7.4 Limitation of Numbers of Choices

Our method selects a few texts as the candidates for a reply based on scores of texts in each text collection (Glossary, Help texts, and Support KB).

It sorts texts in order of their scores, and then selects top n texts as the candidates, provided that all of their scores are not less than t. If multiple texts have a same ranking

Table 2.4: Parameters for selecting candidates for a reply

Text collection	n	t
Glossary	2	0.8
Help texts	5	0.3
Support KB	10	0.1
<UQ> of Dialog cards (see Section 4.3)	1	0.8

in front and behind nth, all of them are selected. Parameters n, t are set for each text collection, as shown in Table 2.4.

If candidates are selected from more than one text collections, they are shown to the user in order of Glossary, Help texts, and Support KB.

2.8 Evaluation and Discussion

To evaluate the performance of introducing the proposed methods, we prepared testsets, and defined a measure for evaluating the output (ranked texts).

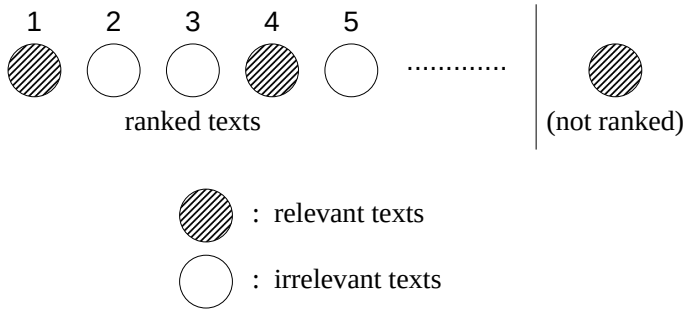
2.8.1 Testsets

The testsets were prepared as follows:

1. We randomly selected some user questions collected by *Dialog Navigator* (see Chapter 5) from each week of November 2002 - September 2003. As a result, 1,290 user questions were selected.

2. A subject gave relevant texts to each user question, based on exhaustive text retrieval through the text collections. Note that some questions have more than one relevant text, and other questions have no relevant texts.

As a result, two testsets were prepared: 163 user questions which have relevant texts of Help texts; and 773 user questions which have relevant texts of Support KB. We did not prepare the testset for Glossary, because almost all of user questions which have relevant texts of Glossary successfully matched the texts without any sophisticated methods. The user questions which have no relevant texts were excluded for the evaluation.



$$\epsilon = \frac{1/1 + 1/4}{1/1 + 1/2 + 1/3} = 0.68$$

Figure 2.11: Calculation of ϵ

2.8.2 Evaluation Rate

We defined the evaluation rate ϵ of the output (ranked texts) for each user question as follows:

$$\epsilon = \frac{\sum_{i \in \mathcal{R}} \frac{1}{i}}{\sum_{j \in \{1,\cdots,n\}} \frac{1}{j}} \tag{2.13}$$

where n is the number of the relevant texts for a user question, and $\mathcal{R}$ is the set of ranks of the relevant texts which our method outputted. Figure 2.11 shows a calculation example of ϵ. Note that this measure is an extension of MRR (mean reciprocal rank) for evaluating open-domain QA systems. Usually, a question for an open-domain QA task has only one answer, but a question for Dialog Navigator often has several answers (relevant texts). Therefore, we introduced the normalization factor as shown in Equation 2.13.

Each method is evaluated by calculating $\overline{\epsilon}$, the average of ϵ on a testset:

$$\overline{\epsilon} = \frac{\epsilon}{N} \tag{2.14}$$

where N is the number of user questions in the testset.

2.8.3 Experiments on Testsets

First, to determine the optimum value for m, *i.e.,* weighting on modifier-head relations, we increased m from 0 to 3.0, and calculated $\overline{\epsilon}$, the average of ϵ, for each m on each testset.

Figure 2.12 shows the results. Generally, $\overline{\epsilon}$ was improved the most around $m = 1.0$: $\overline{\epsilon}$ was gradually improved from $m = 0$ to 1.0; in contrast, $\overline{\epsilon}$ was gradually worsened beyond $m = 1.4$. The best improvement was about 0.06 for Help texts, and about 0.025 for Support KB.

Based on the above results, we decided to fix m at 1.0 in the following evaluations.

Next, we evaluated effectiveness of the other proposed methods, that is to say, the following methods:

nflag Assignment of negation flags (Subsection 2.3.3).

final Removal of final expressions (Subsection 2.3.5).

syn Expression gap resolution by synonymous expression dictionary (Subsection 2.4.1).

question Selection by question types (Subsection 2.6.1).

product Selection by product names (Subsection 2.6.2).

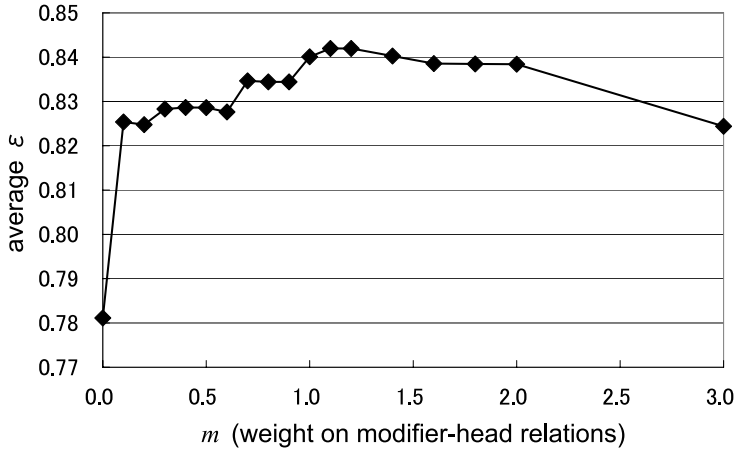
To evaluate the effectiveness, we experimented with each condition in which some of the above methods were disabled, using the testsets. Specifically, we calculated $\overline{\epsilon}$ on each testset, for each of the following conditions:

- **baseline**: none of the above methods were used
- **baseline**+x: only one of the above methods x was used
- **all**: all of the above methods were used
- **all**$-x$: all of the above methods except x were used

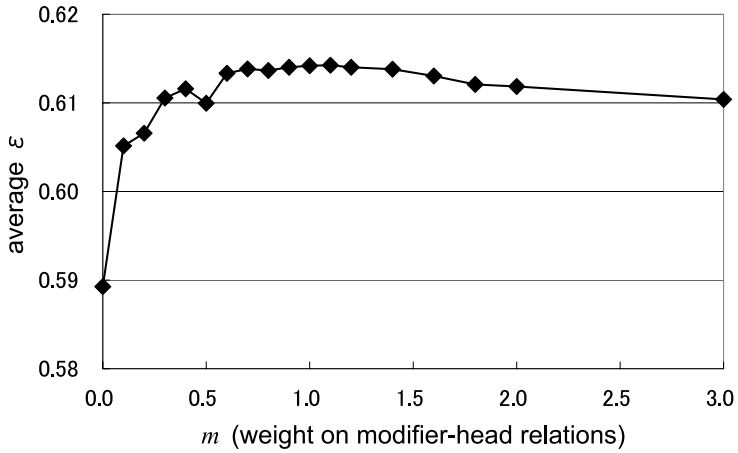
where $x \in \{\textbf{nflag}, \textbf{final}, \textbf{syn}, \textbf{question}, \textbf{product}\}$. In each condition, m was fixed to 1.0.

Table 2.5 and Table 2.6 show the results. Totally, $\overline{\epsilon}$ was significantly improved on both the testsets: the total improvement of $\overline{\epsilon}$ on Help texts was 0.238, and that on Support KB was 0.117.

Among the methods, **syn** improved $\overline{\epsilon}$ the most on both testsets. The other methods (**nflag**, **final**, **question**, and **product**) also gave some improvement on $\overline{\epsilon}$; all of these methods improved $\overline{\epsilon}$ on Support KB, while only **final** improved $\overline{\epsilon}$ on Help Texts. This was because: Help texts had only *how* type texts; there were no *negative expressions* in each Help text; and every Help text had *product names*.



(Help texts: 163 user questions)



(Support KB: 773 user questions)

Figure 2.12: Evaluation of the weighting on M-H relations

Table 2.5: Evaluation of the matching methods (Help texts)

condition	$\overline{\epsilon}$	improvement against **baseline**	improvement against **all**
baseline	0.602		−0.238
baseline + **nflag**	0.602		−0.238
baseline + **question**	0.602		−0.238
baseline + **product**	0.602		−0.238
baseline + **final**	0.605	+0.003	−0.235
baseline + **syn**	0.838	+0.236	−0.002
all − **syn**	0.605	+0.003	−0.235
all − **final**	0.838	+0.236	−0.002
all − **nflag**	0.840	+0.238	
all − **question**	0.840	+0.238	
all − **product**	0.840	+0.238	
all	0.840	+0.238	

(163 user questions; $m = 1.0$)

Table 2.6: Evaluation of the matching methods (Support KB)

condition	$\overline{\epsilon}$	improvement against **baseline**	improvement against **all**
baseline	0.497		−0.117
baseline + **question**	0.498	+0.001	−0.116
baseline + **final**	0.508	+0.011	−0.106
baseline + **nflag**	0.512	+0.015	−0.102
baseline + **product**	0.519	+0.022	−0.095
baseline + **syn**	0.561	+0.064	−0.053
all − **syn**	0.524	+0.027	−0.090
all − **nflag**	0.591	+0.094	−0.023
all − **product**	0.592	+0.095	−0.022
all − **final**	0.604	+0.107	−0.010
all − **question**	0.612	+0.115	−0.002
all	0.614	+0.117	

(773 user questions; $m = 1.0$)

2.8.4 Discussion on Matching Failures

As we have mentioned in Chapter 1, precise and flexible matching of user questions with texts is a critical issue for resolving the various gaps between user questions and texts. To obtain better performance of the matching, we analyzed some of matching failures, *i.e.,* user questions that had small ϵ in the testset. As a result, we found that the matching failures were mainly caused by the following factors:

(a) *Insufficiency of the synonymous expression dictionary.*

As shown in Table 2.5 and Table 2.6, expression gap resolution by the synonymous expression dictionary greatly improved the performance. However, there is still room for further improvement. We found that some of expression gaps were not resolved by the dictionary:

(2.15) Q : 文字を 大きくしたい
moji-wo *ōkiku-shitai*
character-acc enlarge-(want to)
'(I) want to enlarge characters'

A : 文字の サイズを 変更する
moji-no *saizu-wo* *henkō-suru*
character-gen size-acc change-(do)
'change size of characters'

(2.16) Q : *Windows*を 終了する
Windows-wo *shūryō-suru*
Windows-acc shut down-(do)
'shut down Windows'

A : コンピューターの 電源を オフに する
konpyūtā-no *dengen-wo* *ofu-ni* *suru*
computer-gen power supply-acc off-dat (do)
'turn off the power supply of the computer'

In each of above examples, the user question (Q) did not match the relevant text (A). Based on analysis of such matching failures in the question logs for Dialog Navigator, we have continuously revised the synonymous expression dictionary.

(b) *Trivial matches.*

Sometimes user questions matched irrelevant texts because the questions matched fragments which were not so important in the texts:

(2.17) Q : フォントを 追加する
fonto-wo *tsuika-suru*
font-acc add-(do)
'add fonts'

A : フォントを *Windows ME*に インストールせずに フォントを
fonto-wo *Windows ME-ni* *insutōru-sezuni* *font-wo*
font-acc Windows ME-dat install-(without) font-acc
追加する アプリケーションソフトウェアを 使用している 場合、
tsuika-suru *apurikēshonsofutowea-wo* *shiyō-shiteiru* *baai,*
add-(do) application software-acc use-(doing) when
*TrueType*フォントキャッシュの 内容が 破壊される
TrueType-fontokyasshu-no *naiyō-ga* *hakai-sareru*
TrueType font cache-gen content-nom break-(be done)
'When you use an application software which adds fonts without installing the fonts in Windows ME, the contents of TrueType font cache are broken'

In Japanese, mostly, the final part of each sentence is the most important, whereas other parts are less important. As for (2.17), the matched part "フォントを追加する" (*fonto-wo tsuika-suru*, add fonts) is not located on the final, and is trivial for the whole sentence.

To prevent such irrelevant texts from being matched, the matching algorithm should attach special importance to the final part of each sentence.

(c) *Side effects of the weighting on modifier-head relations.*

As shown in Figure 2.12, the weighting on modifier-head relations m significantly improved $\bar{\epsilon}$. However, sometimes the weighting worsened the performance. We examined the side effects, and found that they were mainly caused by the following factors:

(c-1) Parse errors.

Sometimes incorrect modifier-head relations detected by KNP led to matches of irrelevant texts.

(c-2) Zero anaphora.

In Japanese, arguments of verbs are very often omitted. As for Q of (2.18), the object case of "開く" (*hiraku*, open) is omitted: a *zero pronoun* is in the case,

and the pronoun indicates “ファイル” (*fairu*, file). As a result, the modifier-head relation between “ファイル” (*fairu*, file) and “開く” (*hiraku*, open) is not detected, and Q does not match A at a high score if m is large.

(2.18) Q : ファイルを クリックして 開く
fairu-wo *kurikku-shite* *hiraku*
file-acc click-(do) open
‘click a file, and open it’

A : ファイルを 開く
fairu-wo *hiraku*
file-acc open
‘open a file’

(c-3) Modifier-head relation gaps caused by metonymies.

Consider the following example:

(2.19) Q : *GIF*を 表示する
GIF-wo *hyōji-suru*
GIF-acc display-(do)
‘display a GIF’

A : *GIF*の 画像を 表示する
GIF-no *gazō-wo* *hyōji-suru*
GIF-gen image-acc display-(do)
‘display a GIF image’

Q and A are almost synonymous, but the matching score is very small when m is large, because the modifier-head relation between *“GIF”* and “表示する” (*hyōji-suru*, display) in Q does not exist in A. Such matching failures were often observed in the testset.

Q is regard as a metonymy, that is to say, a figure of speech in which the name of one thing is substituted for that of something to which it is related [28]; and A is regarded as the interpretation of the metonymy.

Among those factors, (c) requires our methods to be improved the most. We think that (c) will be resolved by further advancements of NLP techniques: (c-1) will be reduced by refinements of parsers; the resolution of (c-2) requires case analysis; and (c-3) will be resolved by processing of metonymy. Chapter 3 will propose a method to cope with (c-2), based on automatic acquisition of pairs of metonymic expressions and their interpretative expressions, and its application to the resolution of the modifier-head relation gaps.

2.9 Related Work

This section compares our methods with previous studies on application of NLP-oriented techniques for information retrieval in a broad sense (including text retrieval and question answering), with the following focuses:

(i) *precise* matching methods based on full parsing, and

(ii) *flexible* matching methods for resolving expression gaps.

2.9.1 Matching Methods based on Full Parsing

As we have already pointed out at the beginning of this chapter, open-domain QA systems require NLP-oriented matching methods based on sentence structures of both user questions and texts. Importance of such methods for open-domain QA systems is incomparably greater than that for text retrieval systems.

Some of our methods, *e.g.* the full-parsing-based matching methods and selection by question types, were also employed by participant systems of TREC QA Track and NTCIR QAC. For instance, Harabagiu *et al.* [12] and Kawahara *et al.* [34] employed matching methods based on predicate-argument structures. Also, correlation between question types and named entities (*e.g. Who is ...*: person names, *Where is ...*: place names, and *How tall ...*: length) has been generally used.

Strictly speaking, the goal of our methods differs from that of open-domain QA systems, because it is intended to find not "exact" answers A for each user question Q, but the parts Q' which correspond to Q. However, the NLP-oriented matching methods for our goal are as important as those for the open-domain QA systems. In most cases, each answer A and Q' is in a same text, and therefore detection of the parts of Q' directly leads to specification of A. Furthermore, to make asking-backs for vague questions, the detection of Q' is required as we have mentioned in Section 2.1. The neighborhoods of Q' are suitable for such asking-backs, because they make the difference among matched texts more clear. Chapter 4 will describe detailed methods for making such asking-backs.

2.9.2 Matching Methods for Resolving Expression Gaps

The most ordinary method for resolving expression gaps on text retrieval systems is query expansion based on thesauri such as Roget's [35] and WordNet [36]. Salton [10] evaluated

effectiveness of using a thesaurus on SMART system, and showed that it significantly improved performance on text retrieval. Also, some participant at TREC-5 NLP Track [32] reported that query expansion improved performance of their systems.

Furthermore, some QA systems employed thesauri or synonym dictionaries for resolving the expression gaps. For example, FAQ Finder [8] used WordNet, and Dialog Help System of CIMS at Kyoto University [26] used a domain ontology which defined relations between concepts or issues.

However, as we have mentioned in Section 2.4, some expression gaps on phrase levels such as (2.1) and (2.2) are not resolved well only with synonyms, that is, on keyword levels. It is important to resolve such expression gaps on phrase levels, especially when novice users are targeted: they are usually unfamiliar with accurate expressions as used in manuals, so they often use informal expressions.

Several methods for resolving the phrase-level expression gaps have been studied. Kurohashi *et al.* [37] proposed a bottom-up approach for resolving such gaps based on some matching rules, definitions in a Japanese-language dictionary, and a thesaurus [38]. Recently, *paraphrasing*-based methods have been actively studied [39,40]. However, those methods employed on-line strategies, so it is not applicable for large text collections because of high calculation costs.

To cope with the problem, we proposed an off-line strategy for resolving phrase-level expression gaps, which is applicable for large text collections. The method can resolve such gaps with low calculation cost, by using a relatively small domain-dependent dictionary for synonymous phrases.

However, some expression gaps are not resolved well by our method. There are two major types of such gaps: *zero anaphora* and *metonymy*. As we have mentioned in Subsection 2.8.4, these gaps would raise side effects of the weighting on modifier-head relations. Next chapter will target one of these types, that is, *metonymy* type.

2.10 Summary of this Chapter

In this chapter, we proposed several methods for realizing precise and flexible matching for a user question with texts:

- Basic methods:
  - parsing and keyword extraction

  - unification of *bunsetsu*
  - assignment of negation flags
- Methods for precise matching:
  - weighting on modifier-head relations
  - selection by question types
  - selection by product names
- Methods for flexible matching:
  - synonymous expression dictionary
  - ontological dictionary
  - removal of final expressions

The above methods can be applied for real-world text collections.

And then, we showed the effectiveness of the proposed methods by testsets, and discussed advantage and limitation of our methods. In the next chapter, we proposed an extension of those methods for resolving the limitation: matching of metonymic expressions, which are very important phenomena of natural language.

Chapter 3

Matching Methods for Metonymic Expressions

3.1 Introduction

As we have mentioned in the previous chapter, it is critical for a text-based QA system to match a question with texts precisely. In order to achieve precise matching, recent systems for question-answering are based on full-parsing of user questions and texts.

In practice, however, when a user uses a metonymy in his/her question, the matching with texts based on full-parsing often fails. Metonymy is a figure of speech in which the name of one thing is substituted for that of something to which it is related [28]:

(3.1) The ham sandwich is waiting for his check.

In this example sentence, "the ham sandwich" indicates "the man who ordered a ham sandwich". Metonymies are also used in the domain of personal computers:

(3.2) *Word*を 開く
Word-wo *hiraku*
Word-acc open
'open a Word'

(3.3) *Word*の ファイルを 開く
Word-no *fairu-wo* *hiraku*
Word-gen file-acc open
'open a Word file'

(3.4) 電源を 入れる
dengen-wo *ireru*
power supply-acc turn on
'turn the power supply on'

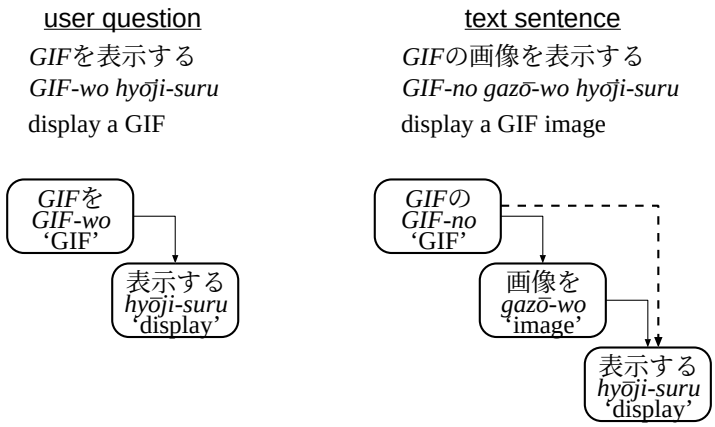


Figure 3.1: A matching failure because of a metonymy

(3.5) 電源 スイッチを 入れる
dengen *suicchi-wo* *ireru*
power supply switch-acc turn on
'turn the power supply switch on'

In the above examples, (3.2) can be interpreted to (3.3), and (3.4) can be interpreted to (3.5). Namely, *"Word"* indicates *"Word のファイル"* (*Word-no fairu* ,Word file), and "電源" (*dengen*, power supply) indicates "電源スイッチ" (*dengen suicchi*, power supply switch).

Figure 3.1 shows an example of matching failure. In this example, the user seems to use *"GIF"* as a metonymy of *"GIF の画像"* (*GIF-no gazō*, GIF image). If the full-parsing based matching method is used, this matching score will be lower, because the direct relation between *"GIF"* and "表示する" (*hyōji-suru*, display) in the user question does not exist in the text sentence. As this example shows, metonymic expressions could raise gaps of syntactic structures.

The above problem is so acute for QA systems for novice users, such as Dialog Navigator, because novice users often use metonymy in their question for the following reasons:

- their motivation for making short questions is relatively larger compared with that of experts, and
- they often use abrupt expressions that their acquaintances do, because they are unfamiliar with accurate expressions used in manuals.

Moreover, processing metonymy is vital for other NLP tasks such as machine translation [41].

In early 1990's, most previous studies on processing metonymy by computers were based on manually-constructed ontologies, semantic frames, or logical forms [42, 43]. However, such knowledge structures require heavy cost of construction and maintenance, and makes scaling up quite difficult. Therefore, corpus-based approaches on processing metonymy were studied recently [44–46].

As Fass [47] pointed out, processing metonymies involves two main steps: recognition and interpretation. Most previous studies targeted the interpretation process, that is, to find expressions as interpretations for each expression as a metonymy [45, 46]. Namely, they targeted a problem of finding interpretations for each metonymy that is in textbooks. In contrast, the recognition process, that is, to judge whether an expression is a metonymy or not, have almost never studied.

In contrast, for the domain that this thesis targets, that is, question answering for general users of personal computers, both the recognition process and the interpretation process should be targeted, because no knowledge for defining the collection of metonymies exists in the real world.

This chapter proposes a method for automatic acquisition of pairs of metonymic expressions and their interpretative expressions from large corpora, and a method for applying the acquired pairs to resolve the gaps of modifier-head relations as shown in Figure 3.1. The corpora include the text collections provided by Microsoft Corporation (Table 2.1), and a lot of user questions collected by our system *Dialog Navigator* (see Chapter 5). First, Section 3.2 defines *metonymic expressions* and *interpretative expressions* that this thesis targets. Next, Section 3.3 gives the method for automatically acquiring pairs of metonymic expressions and their interpretative expressions from large corpora, and Section 3.4 describes a method for applying the acquired pairs to resolve the gaps of modifier-head relations caused by metonymies. And then, Section 3.5 evaluates the proposed methods from the following two aspects: one is whether each acquired metonymic expression is correctly interpreted; and the other is whether the proposed methods work well on the matching a user question with texts, that is, a testset based evaluation. Finally, Section 3.6 compares our methods with those of previous work, and Section 3.7 concludes this chapter.

3.2 Metonymic Expressions and Interpretative Expressions

In this thesis, we target the combination of the following two expressions:

(α) $\boldsymbol{A}\ \boldsymbol{P} \rightarrow \boldsymbol{V}$

(β) $\boldsymbol{A}$ (*の; no*) $\rightarrow \boldsymbol{B}\ \boldsymbol{P} \rightarrow \boldsymbol{V}$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ mean nouns, $\boldsymbol{P}$ means a case-marker, $\boldsymbol{V}$ means a predicate, and "→" means a modifier-head relation[1]. We ignore whether "の" (*no*; genitive case marker) exists or not, because both "$\boldsymbol{A}$ (の; *no*) → $\boldsymbol{B}$" and "$\boldsymbol{A} \rightarrow \boldsymbol{B}$" forms a noun phrase[2]. As for the example of Figure 3.1, $\boldsymbol{A}$=*"GIF"*, $\boldsymbol{B}$="画像" (*gazō*, image), $\boldsymbol{P}$="を" (*wo*; accusative case marker), and $\boldsymbol{V}$="表示" (*hyōji*, display). Namely, (α) is *"GIF* を表示*"* (*GIF-wo hyōji*; display a GIF), and (β) is *"GIF(*の*)*画像を表示*"* (*GIF(-no) gazō-wo hyōji*, display a GIF image). In this case, it seems that (α) is a metonymy, and (β) is its interpretation.

We preliminarily extracted the combinations of (α) and (β) from corpora, and the result shows that most of the combinations are correct as metonymies and their interpretations. Therefore, we tried extraction of the combinations of (α) and (β), as an automatic acquisition of metonymies and their interpretations.

In order to get a lot of metonymic expressions automatically, we use huge corpora: the text collections provided by Microsoft Corporation (Table 2.1), and a lot of user questions collected by Dialog Navigator (see Chapter 5) and *Hanashi-kotoba Kensaku*[3]. Most of the user questions are inputted by novice users, so they include plenty of metonymies.

In the following sections, we call (α) as *metonymic expression*, and (β) as *interpretative expression*.

3.3 Acquisition of Metonymic Expressions and their Interpretative Expressions

From parse results of all sentences in the corpora, our method automatically acquires pairs of metonymic expressions and their interpretative expressions as follows:

[1] Japanese is a head-final language, and the arguments of each predicate are placed left to the predicate.
[2] "の" (*no*) is similar to the English preposition 'of', but has more meanings.
[3] `http://www.microsoft.com/japan/enable/nlsearch/`

1. Collect candidates of metonymic expressions (C_α): every phrase which matches the pattern "$\boldsymbol{A_\alpha}$ $\boldsymbol{P_\alpha} \rightarrow \boldsymbol{V_\alpha}$" is collected, as far as frequency f_α is not less than the threshold t_α, that is, $f_\alpha \geq t_\alpha$.

2. Collect candidates of interpretative expressions (C_β): every phrase which matches the pattern "$\boldsymbol{A_\beta}$ *(no)* $\rightarrow$ $\boldsymbol{B_\beta}$ $\boldsymbol{P_\beta} \rightarrow \boldsymbol{V_\beta}$" is collected, as far as frequency f_β is not less than the threshold t_β, that is, $f_\beta \geq t_\beta$.

3. For each metonymic expression of C_α, find its interpretative expressions in which $\boldsymbol{A_\beta} = \boldsymbol{A_\alpha}$, $\boldsymbol{P_\beta} = \boldsymbol{P_\alpha}$ and $\boldsymbol{V_\beta} = \boldsymbol{V_\alpha}$ from C_β.

where $\boldsymbol{A_\alpha}$, $\boldsymbol{A_\beta}$ and $\boldsymbol{B_\beta}$ mean nouns, $\boldsymbol{P_\alpha}$ and $\boldsymbol{P_\beta}$ mean case-markers, and $\boldsymbol{V_\alpha}$ and $\boldsymbol{V_\beta}$ mean predicates, and "$\rightarrow$" means a modifier-head relation. In the following paragraphs, we omit the indication of "$\rightarrow$" to simplify the descriptions of metonymic expressions and interpretative expressions. In addition, to improve readability of the expressions, we insert "の" (*no*, genitive case marker) according to circumstances.

To avoid acquiring incorrect expressions, out method introduces the thresholds of frequency (t_α and t_β). We experimentally set the thresholds of frequency $t_\alpha = t_\beta = 3$. In addition, our method excludes the following expressions when it collects C_α and C_β:

- Expressions in which $\boldsymbol{A_\alpha}$ or $\boldsymbol{A_\beta}$ are modified by left *bunsetsu* are excluded, because they may be parts of bigger noun phrases. Here is an example:

(3.6) デスクトップの 表示を 追加
desukutoppu-no *hyōji-wo* *tsuika*
desktop-gen display-acc add
'add "display of the desktop"'

(3.7) * 表示を 追加
hyōji-wo *tsuika*
display-acc add

(3.8) 表示の 形式を 追加
hyōji-no *keishiki-wo* *tsuika*
display-gen style-acc add
'add a display style'

Note that (3.7) is a substring of (3.6). As for this example, the substring (3.7) matches the pattern for collecting C_α. However, "デスクトップの表示" (*desukutoppu-no hyōji*, display of the desktop) forms a proper noun (*i.e.* an item name in a menu

bar), and the fragment "表示" (*hyōji*, display) does not indicate the proper noun; if (3.7) is collected as a candidate of metonymic expressions, (3.8) will be incorrectly found as its interpretative expression. This condition rules (3.7) out, because the *bunsetsu* "表示を" (*hyōji-wo*, display-acc) is modified by left *bunsetsu* "デスクトップの" (*desukutoppu-no*, desktop-gen).

- Expressions in which either of parentheses (*i.e.* 「, 」, (,), <, >, [, and]) occurs are excluded, because occurrences of these parentheses implies that the expressions forms proper noun phrases in many cases. Usually, such expressions have no connection with metonymy, and collecting them often leads to incorrect interpretations of metonymic expressions just as collecting (3.7) does. For instance, the following expression is ruled out from C_α:

  (3.9) ［検索］を 削除
  [kensaku]-wo *sakujo*
  [search]-acc remove
  'remove [search] (an item name)'

- Expressions in which either of $\boldsymbol{V_\alpha}$ or $\boldsymbol{V_\beta}$ is in passive or causative voice are excluded. This condition is based on a preliminary examination. For example, the following expression is ruled out:

  (3.10) アプリケーションが 発生させる
  apurikēshon-ga *hassei-saseru*
  application-nom cause-(causative voice)
  'The application causes ...'

- Expressions which have long distance modifier-head relations are excluded. Such relations possibly involve parse errors, which may result in acquiring incorrect expression pairs. Namely, if either of modifier-head relations in an expression has a distance of not less than three *bunsetsu*, that is, the relation puts not less than two *bunsetsu* between, the expression is excluded.

We applied the method for 1,351,981 sentences in the corpora, including 762,353 user questions and 589,628 sentences in the text collections. As a result, we got 1,126 pairs of metonymic expressions and their interpretative expressions. Table 3.1∼3.5 shows the examples. In these tables, α_i are metonymic expressions, and $\beta_{i,j}$ are interpretative expressions (the right columns of "pair evaluation", "$\boldsymbol{B_\gamma}$", and "group evaluation" will be

Table 3.1: Acquired metonymic expressions, their interpretations, and evaluation (1)

	$\boldsymbol{A_\alpha}$ / $\boldsymbol{A_\beta}$	$\boldsymbol{B_\beta}$	$\boldsymbol{P_\alpha}$ / $\boldsymbol{P_\beta}$	$\boldsymbol{V_\alpha}$ / $\boldsymbol{V_\beta}$	f_α / f_β	pair evaluation	$\boldsymbol{B_\gamma}$	group evaluation
α_1	エラー		が	出る	1,681			**A**
	error		nom	be caused				
	erā		*ga*	*deru*				
	'an error is caused'							
$\beta_{1,1}$	エラー	表示	が	出る	68	**correct**		
	erā	*hyōji*	*ga*	*deru*				
	error	indication	nom	be caused				
	'an error indication is caused'							
$\beta_{1,2}$	エラー	報告	が	出る	9	**correct**		
	erā	*hōkoku*	*ga*	*deru*				
	error	report	nom	be caused				
	'an error report is caused'							
$\beta_{1,3}$	エラー	画面	が	出る	6	**correct**		
	erā	*gamen*	*ga*	*deru*				
	error	screen	nom	be caused				
	'an error screen is caused'							
$\beta_{1,4}$	エラー	情報	が	出る	4	**correct**		
	erā	*jōhō*	*ga*	*deru*				
	error	information	nom	be caused				
	'an error information is caused'							
$\beta_{1,5}$	エラー	メッセージ	が	出る	3	**correct**		
	erā	*messēji*	*ga*	*deru*				
	error	message	nom	be caused				
	'an error message is caused'							
$\beta_{1,6}$	エラー	署名	が	出る	3	**correct**		
	erā	*shomei*	*ga*	*deru*				
	error	signature	nom	be caused				
	'an error signature is caused'							
α_2	電源		を	入れる	290			**A**
	dengen		*wo*	*ireru*				
	power supply		acc	turn on				
	'turn the power supply on'							
$\beta_{2,1}$	電源	スイッチ	を	入れる	5	**correct**		
	dengen	*suicchi*	*wo*	*ireru*				
	power supply	switch	acc	turn on				
	'turn the power supply switch on'							

described in the next section). The result was plenty of interesting examples: *i.e.* $\alpha_2 - \beta_{2,1}$ and $\alpha_{10} - \beta_{10,1}$.

In addition, we examined the contribution of each corpus for acquiring the pairs. Namely, we applied the proposed method to either the user questions or the text collections respectively, and counted the number of the acquired pairs (Figure 3.2). The result suggests that the user questions have more metonymic expressions and interpretative expressions than the text collections. This is a promising result, because it implies that more pairs would be acquired if more user questions are collected by continuing the operation of the systems.

Table 3.2: Acquired metonymic expressions, their interpretations, and evaluation (2)

	$\boldsymbol{A_\alpha}$ / $\boldsymbol{A_\beta}$	$\boldsymbol{B_\beta}$	$\boldsymbol{P_\alpha}$ / $\boldsymbol{P_\beta}$	$\boldsymbol{V_\alpha}$ / $\boldsymbol{V_\beta}$	f_α / f_β	pair evaluation	$\boldsymbol{B_\gamma}$	group evaluation
α_3	元		に	戻る	276			**A**
	moto		*ni*	*modoru*				
	former		dat	turn back				
	'turn back to the former'							
$\beta_{3,1}$	元 の	サイズ	に	戻る	12	**correct**		
	moto	*saizu*	*ni*	*modoru*				
	former	size	dat	turn back				
	'turn back to the former size'							
$\beta_{3,2}$	元 の	設定	に	戻る	10	**correct**		
	moto	*settei*	*ni*	*modoru*				
	former	setting	dat	turn back				
	'turn back to the former setting'							
$\beta_{3,3}$	元 の	ページ	に	戻る	5	**correct**		
	moto	*pēji*	*ni*	*modoru*				
	former	page	dat	turn back				
	'turn back to the former page'							
$\beta_{3,4}$	元 の	位置	に	戻る	4	**correct**		
	moto	*ichi*	*ni*	*modoru*				
	former	position	dat	turn back				
	'turn back to the former position'							
$\beta_{3,5}$	元 の	画面	に	戻る	3	**correct**		
	moto	*gamen*	*ni*	*modoru*				
	former	screen	dat	turn back				
	'turn back to the former screen'							
$\beta_{3,6}$	元 の	状態	に	戻る	3	**correct**		
	moto	*jōtai*	*ni*	*modoru*				
	former	status	dat	turn back				
	'turn back to the former status'							
$\beta_{3,7}$	元 の	値	に	戻る	3	**correct**		
	moto	*atai*	*ni*	*modoru*				
	former	value	dat	turn back				
	'turn back to the former value'							
$\beta_{3,8}$	元 の	表示	に	戻る	3	**correct**		
	moto	*hyōji*	*ni*	*modoru*				
	former	indication	dat	turn back				
	'turn back to the former indication'							
α_4	印刷		を	実行	141			**C**
	insatsu		*wo*	*jikkō*				
	printout		acc	execute				
	'execute a printout'							
$\beta_{4,1}$	印刷	プレビュー	を	実行	12	**incorrect**		
	insatsu	*prebyū*	*wo*	*jikkō*				
	printout	preview	acc	execute				
	'execute a printout preview'							
$\beta_{4,2}$	印刷	ジョブ	を	実行	4	**correct**		
	insatsu	*jobu*	*wo*	*jikkō*				
	printout	job	acc	execute				
	'execute a printout job'							
$\beta_{4,3}$	印刷	処理	を	実行	4	**correct**		
	insatsu	*shori*	*wo*	*jikkō*				
	printout	process	acc	execute				
	'execute a printout process'							
$\beta_{4,4}$	印刷	コマンド	を	実行	3	**correct**		
	insatsu	*komando*	*wo*	*jikkō*				
	printout	command	acc	execute				
	'execute a printout command'							

Table 3.3: Acquired metonymic expressions, their interpretations, and evaluation (3)

	A_α / A_β	B_β	P_α / P_β	V_α / V_β	f_α / f_β	pair evaluation	B_γ	group evaluation
α_5	動作		が	遅い	123			**A**
	dōsa		*ga*	*osoi*				
	movement		nom	slow				
	'movement is slow'							
$\beta_{5,1}$	動作	速度	が	遅い	8	**correct**		
	dōsa	*sokudo*	*ga*	*osoi*				
	movement	speed	nom	slow				
	'movement speed is slow'							
α_6	プログラム		を	起動	107		ϕ	**E**
	puroguramu		*wo*	*kidō*				
	program		acc	launch				
	'launch a program'							
$\beta_{6,1}$	プログラム の 削除		を	起動	4	**incorrect**		
	puroguramu	*sakujo*	*wo*	*kidō*				
	program	uninstall	acc	launch				
	'launch " uninstall a program"'							
α_7	文字		が	ずれる	97			**A**
	moji		*ga*	*zureru*				
	character		nom	slip off				
	'characters slip off'							
$\beta_{7,1}$	文字 の 位置		が	ずれる	19	**correct**		
	moji	*ichi*	*ga*	*zureru*				
	character	position	nom	slip off				
	'a position of characters slips off'							
$\beta_{7,2}$	文字 の 間隔		が	ずれる	4	**correct**		
	moji	*kankaku*	*ga*	*zureru*				
	character	spacing	nom	slip off				
	'spacings of characters slip off'							
$\beta_{7,3}$	文字	列	が	ずれる	3	**correct**		
	moji	*retsu*	*ga*	*zureru*				
	character	string	nom	slip off				
	'a character string slips off'							
α_8	画像		を	挿入	69			**A**
	gazō		*wo*	*sōnyū*				
	image		acc	insert				
	'insert an image'							
$\beta_{8,1}$	画像	ファイル	を	挿入	6	**correct**		
	gazō	*fairu*	*wo*	*sōnyū*				
	image	file	acc	insert				
	'insert an image file'							
α_9	ファイル		が	破損	56		内容	**B**
	fairu		*ga*	*hason*			*naiyō*	
	file		nom	be broken			'content'	
	'a file is broken'							
$\beta_{9,1}$	ファイル の 一部		が	破損	3	**correct**		
	fairu	*ichibu*	*ga*	*hason*				
	file	part of	nom	be broken				
	'a part of a file is broken'							
α_{10}	改行		が	変わる	34			**A**
	kaigyō		*ga*	*kawaru*				
	line feed		nom	change				
	'a line feed changes'							
$\beta_{10,1}$	改行 の 幅		が	変わる	3	**correct**		
	kaigyō	*haba*	*ga*	*kawaru*				
	line feed	width	nom	change				
	'a width of a line feed changes'							

Table 3.4: Acquired metonymic expressions, their interpretations, and evaluation (4)

	A_α / A_β	B_β	P_α / P_β	V_α / V_β	f_α / f_β	pair evaluation	B_γ	group evaluation
α_{11}	*JPG*		で	保存	20			**A**
	JPG		*de*	*hozon*				
	JPG		ins	save				
	‘save ... using JPG’							
$\beta_{11,1}$	*JPG*	形式	で	保存	13	**correct**		
	JPG	*keishiki*	*de*	*hozon*				
	JPG	format	ins	save				
	‘save ... using JPG format’							
α_{12}	画面		が	変わる	18		背景	**B**
	gamen		*ga*	*kawaru*			*haikei*	
	screen		nom	change			‘background’	
	‘a screen changes’							
$\beta_{12,1}$	画面	の 解像度	が	変わる	7	**correct**		
	gamen	*kaizōdo*	*ga*	*kawaru*				
	screen	resolution	nom	change				
	‘screen resolution changes’							
$\beta_{12,2}$	画面	の 色	が	変わる	4	**correct**		
	gamen	*iro*	*ga*	*kawaru*				
	screen	color	nom	change				
	‘screen color changes’							
$\beta_{12,3}$	画面	の サイズ	が	変わる	3	**correct**		
	gamen	*saizu*	*ga*	*kawaru*				
	screen	size	nom	change				
	‘screen size changes’							
$\beta_{12,4}$	画面	の 表示	が	変わる	3	**correct**		
	gamen	*hyōji*	*ga*	*kawaru*				
	screen	display	nom	change				
	‘screen display changes’							
α_{13}	ドメイン		を	追加	7		ϕ	**E**
	domein		*wo*	*tsuika*				
	domain		acc	add				
	‘add a domain (name)’							
$\beta_{13,1}$	ドメイン	ユーザ	を	追加	3	**incorrect**		
	domein	*yūzā*	*wo*	*tsuika*				
	domain	user	acc	add				
	‘add a domain user’							
α_{14}	アドレス		を	開く	4		ϕ	**E**
	adoresu		*wo*	*hiraku*				
	address		acc	open				
	‘open an address’							
$\beta_{14,1}$	アドレス	帳	を	開く	43	**incorrect**		
	adoresu	*chō*	*wo*	*hiraku*				
	address	book	acc	open				
	‘open an address book’							
$\beta_{14,2}$	アドレス	帖	を	開く	3	**incorrect**		
	adoresu	*chō*	*wo*	*hiraku*				
	address	book	acc	open				
	‘open an address book’							
α_{15}	ワード		が	消える	4		プログラム	**D**
	Wādo		*ga*	*kieru*			*puroguramu*	
	Word		nom	disappear			‘program’	
	‘Word disappears’						アイコン	
$\beta_{15,1}$	ワード	の メニュー	が	消える	4	**correct**	*aikon*	
	Wādo	*menyū*	*ga*	*kieru*			‘icon’	
	Word	menu	nom	disappear				
	‘the menu of Word disappears’							
$\beta_{15,2}$	ワード	の フォント	が	消える	4	**incorrect**		
	Wādo	*fonto*	*ga*	*kieru*				
	Word	font	nom	disappear				
	‘a font of Word disappears’							

Table 3.5: Acquired metonymic expressions, their interpretations, and evaluation (5)

	$\boldsymbol{A_\alpha}$ / $\boldsymbol{A_\beta}$	$\boldsymbol{B_\beta}$	$\boldsymbol{P_\alpha}$ / $\boldsymbol{P_\beta}$	$\boldsymbol{V_\alpha}$ / $\boldsymbol{V_\beta}$	f_α / f_β	pair evaluation	$\boldsymbol{B_\gamma}$	group evaluation
α_{16}	画面		に	従う	3			**A**
	gamen		*ni*	*shitagau*				
	screen		dat	follow				
	'follow the screen'							
$\beta_{16,1}$	画面	の 指示	に	従う	96	**correct**		
	gamen	*shiji*	*ni*	*shitagau*				
	screen	instruction	dat	follow				
	'follow the instruction in the screen'							
$\beta_{16,1}$	画面	の メッセージ	に	従う	3	**correct**		
	gamen	*messēji*	*ni*	*shitagau*				
	screen	message	dat	follow				
	'follow the message in the screen'							
α_{17}	ドキュメント		を	表示	3		内容	**E**
	dokyumento		*wo*	*hyōji*			*naiyō*	
	document		acc	display			'content'	
	'display a document'							
$\beta_{17,1}$	ドキュメント	の 種類	を	表示	5	**incorrect**		
	dokyumento	*shurui*	*wo*	*hyōji*				
	document	type	acc	display				
	'display a document type'							
α_{18}	MO		を	使用	3			**A**
	MO		*wo*	*shiyō*				
	MO		acc	use				
	'use a MO'							
$\beta_{18,1}$	MO	装置	を	使用	4	**correct**		
	MO	*sōchi*	*wo*	shiyō				
	MO	device	acc	use				
	'use a MO device'							

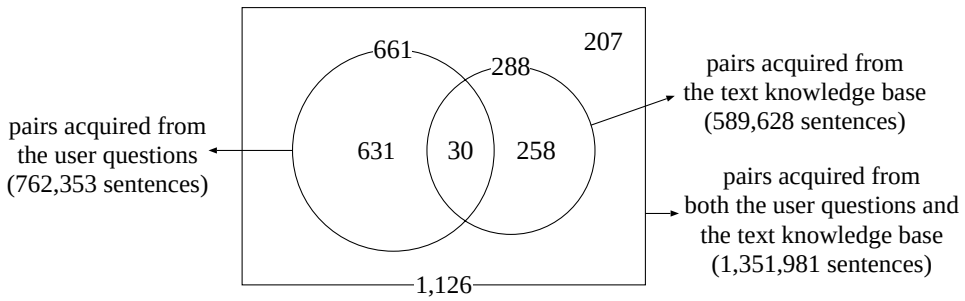


Figure 3.2: Numbers of acquired pairs from each corpus

3.4 Application for Matching

Our method resolves the expression gaps by registering acquired metonymic expressions and their interpretative expressions in the synonymous expression dictionary. For example, by registering *"GIF を表示"* (*GIF-wo hyōji*, display a GIF) and *"GIF の画像を表示"* (*GIF-no gazō-wo hyōji*, display a GIF image) as synonymous expressions, the matching score (Subsection 2.7.1, $m = 1.0$) in Figure 3.1 increases from 0.27 to 1.0 .

Note that the recursive relation expansion for the synonymous expression dictionary (see Subsection 2.4.1) is also applied to the pairs of metonymic expressions and interpretative expressions.

3.5 Evaluation and Discussion

In order to examine our method, we made two kinds of evaluations. One is a judgment whether acquired metonymic expressions are correctly interpreted, and another is the evaluation of the effects on the testsets of Dialog Navigator.

3.5.1 Evaluation on Interpretations of Acquired Metonymic Expressions

We randomly selected pairs of metonymic expressions and their interpretative expressions acquired by the proposed method, and we gave each pair a judgment whether the interpretative expression is correct as an interpretation for the metonymic expression.

First, the 1,126 acquired pairs were divided into groups by each metonymic expression "$\boldsymbol{A_\alpha}$ $\boldsymbol{P_\alpha}$ $\boldsymbol{V_\alpha}$". As a result, they were divided into 847 groups (*metonymic expression groups*). There were two types of the metonymic expressions: 679 metonymic expressions were *single-interpretation expressions*, each of which had only one pair; other 168 metonymic expressions were *multiple-interpretation expressions*, each of which had multiple pairs (totally, the 168 expressions had 447 pairs). For instance, α_2 is a single-interpretation expression, because it had only one pair with $\beta_{2,1}$. In contrast, α_1 is a multiple-interpretation expression, because it had multiple pairs with the interpretative expressions $\beta_{1,1} \sim \beta_{1,6}$.

We randomly selected 200 metonymic expression groups among the 847 groups, including 163 single-interpretation expressions, and 37 multiple-interpretation expressions (corresponding with 101 pairs). Totally, the 200 groups had 264 pairs. After that, we

evaluated each of the 200 groups as follows (examples of the evaluation are shown in the right side of Table 3.1∼3.5):

1. *Pair evaluation.*
   We gave either of the following evaluations to each of the 264 pairs, from the viewpoint whether it is correct as an interpretation of the metonymic expression ("pair evaluation" in Table 3.1∼3.5):

   correct As for the Windows-family environment, one of the situations imagined out of the metonymic expression corresponds with that imagined out of the interpretative expression.

   incorrect The pair does not satisfy the above condition.

   For instance, from α_4, we can imagine a situation that "[print] is selected from a menu bar of an application, and print data is sent for a printer". We evaluated the pairs with $\beta_{4,2}$, $\beta_{4,3}$ and $\beta_{4,4}$ as **correct**, because the situation can also be imagined from $\beta_{4,2}$, $\beta_{4,3}$ and $\beta_{4,4}$. In contrast, we evaluated the pair with $\beta_{4,1}$ as **incorrect**, because the situation imagined from $\beta_{4,1}$, that is, "[print preview] is selected from a menu bar of an application, and a preview window is displayed on the screen", does not correspond with that of α_4.

2. *Listing of other interpretations.*
   If there was any other important interpretative expression "$\boldsymbol{A_\gamma}$ *(の; no)* $\boldsymbol{B_\gamma}$ $\boldsymbol{P_\gamma}$ $\boldsymbol{V_\gamma}$" for each metonymic expression "$\boldsymbol{A_\alpha}$ $\boldsymbol{P_\alpha}$ $\boldsymbol{V_\alpha}$" besides the acquired interpretative expressions, we listed such $\boldsymbol{B_\gamma}$ ("$\boldsymbol{B_\gamma}$" in Table 3.1∼3.5). Furthermore, if we judged that "$\boldsymbol{A_\alpha}$ $\boldsymbol{P_\alpha}$ $\boldsymbol{V_\alpha}$" was not a metonymy, we included "ϕ" in $\boldsymbol{B_\gamma}$. For each metonymic expression, we regarded the collection including both **correct** interpretative expressions and "$\boldsymbol{A_\gamma}$ *(の; no)* $\boldsymbol{B_\gamma}$ $\boldsymbol{P_\gamma}$ $\boldsymbol{V_\gamma}$" as the *collection of genuine interpretative expressions.*

   Specifically, only in the case that there were other major situations for the metonymic expression besides all the situations that could be imagined from the collection of the corresponding interpretative expressions, we listed $\boldsymbol{B_\gamma}$, the interpretative expressions of which expressed the other major situations. For example, as for α_{15}, we could imagine other situations than "Word disappears from the start menu", so we listed $\boldsymbol{B_\gamma}$ as follows:

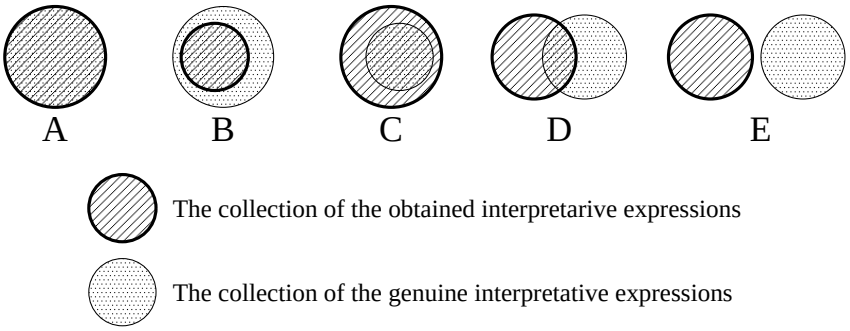


Figure 3.3: Evaluations of metonymic expression groups

- $\boldsymbol{B_\gamma}$=“プログラム” (*puroguramu*, program)
  the situation: “the program files of Word disappear from the hard disk”
- $\boldsymbol{B_\gamma}$=“アイコン” (*aikon*, icon)
  the situation: “the icon of Word disappears from the desktop”

3. *Group evaluation.*
   Based on the above results, we categorized each of the metonymic expression groups into the following evaluations (“group evaluation” in Table 3.1∼3.5). Figure 3.3 shows the relations among the evaluations.

   A Every pairs in the group was evaluated as **correct**, and there were no other major interpretations ($\boldsymbol{B_\gamma}$).

   B Every pairs in the group was evaluated as **correct**, but there were other major interpretations ($\boldsymbol{B_\gamma}$).

   C **correct** and **incorrect** were mixed in the group, and there were no other major interpretations ($\boldsymbol{B_\gamma}$).

   D **correct** and **incorrect** were mixed in the group, and there were other major interpretations ($\boldsymbol{B_\gamma}$).

   E Every pairs in the group was evaluated as **incorrect**.

Table 3.6 shows the results of the pair evaluation, and Table 3.7 shows the results of the group evaluation. More than 80% of the pairs were evaluated as **correct**, and 65% of

Table 3.6: Results of the pair evaluation

evaluation	# of pairs
correct	222 (84%)
incorrect	42 (16%)
total	264 (100%)

Table 3.7: Results of the group evaluation

evaluation	single-interpretation expressions	multiple-interpretation expressions	all
A	102 (63%)	28 (76%)	130 (65%)
B	27 (17%)	3 (8%)	30 (15%)
C	——	3 (8%)	3 (2%)
D	——	1 (3%)	1 (1%)
E	34 (21%)	2 (5%)	36 (18%)
total	163 (100%)	37 (100%)	200 (100%)

the groups were evaluated as **A**. In contrast, the ratios of **C** and **D** were very small, and the ratio of **E** was less than 20%.

Note that the boundaries between **A** and **B**, and between **C** and **D** are essentially ambiguous, because it is very difficult to list all possible interpretations for each metonymic expression. As for α_3, we acquired eight interpretative expressions ($\beta_{3,1} \sim \beta_{3,8}$), but we could also list other interpretative expressions as follows:

(3.11) 元の 場所に 戻る
moto-no *basho-ni* *modoru*
former-gen place-acc turn back
'turn back to the former place'

(3.12) 元の 配置に 戻る
moto-no *haichi-ni* *modoru*
former-gen layout-acc turn back
'turn back to the former layout'

However, we actually did not list any of them, because we judged that the collection of $\beta_{3,1} \sim \beta_{3,8}$ covered all of major situations for α_3.

The above ambiguity problem will be resolved by continuing the operation of the systems, and collecting more user questions. Consider a case that a user question includes (3.11), and a relevant text for the question includes α_3: a matching of the question with the text will fail at first, because of the lack of the pair of α_3 and (3.12) in the synonymous expression dictionary; however, iteration of the matching failure will lead to acquisition of the pair of α_3 and (3.11). Assuming both metonymic expression groups evaluated into **A** and **B** are considered to be interpreted correctly, about 80% of metonymic expressions are given correct interpretations.

3.5.2 Performance on Testsets

To evaluate the performance of introducing metonymic expressions into the matching methods, we prepared testsets. We use the same measure $\overline{\epsilon}$ as described in Subsection 2.8.2.

The testsets were prepared by selecting user questions that satisfies both the following conditions, from the 1,290 user questions mentioned in Subsection 2.8.1:

(a) The user question includes either of the acquired metonymic expressions or the acquired interpretative expressions.

(b) The user question has relevant texts either of Help texts or Support KB.

Condition (a) was satisfied 226 out of the 1,290 user questions, and condition (b) was satisfied 147 out of the 226 user questions. As a result, two testsets were prepared: 31 user questions which have relevant texts of Help texts, and 140 user questions which have relevant texts of Support KB [4].

To evaluate the effectiveness of our method, we experimented with each of the following conditions using the testsets:

baseline Using the methods described in Chapter 1.

metonymy Using the methods described in Chapter 1, and incorporated the acquired metonymic expressions and their interpretative expressions into *synonymous expression dictionary.*

[4] 24 out of the 147 user questions have relevant texts of both Help texts and Support KB.

We increased the weight on M-H relations m from 0 to 3.0, and calculated $\overline{\epsilon}$ for each condition.

Figure 3.4 shows the overall result, and Table 3.8 shows the number of questions on which ϵ was improved or worsened, where $m = 1.0$. The result indicates that introducing metonymic expressions significantly improves the performance. In addition, it also shows the effectiveness of weighting on M-H relations.

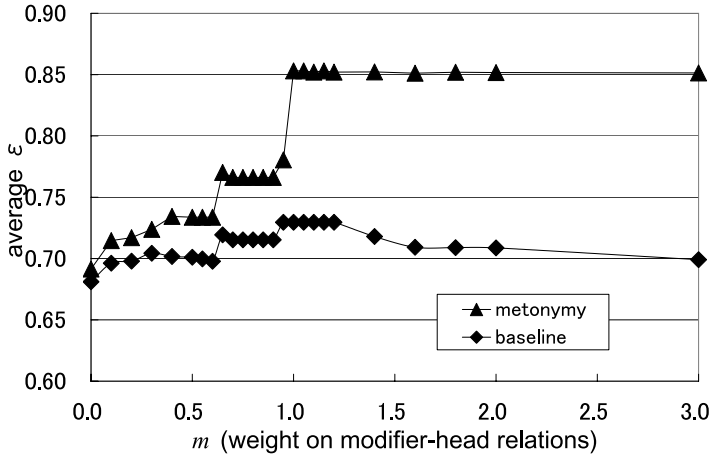
Table 3.9 shows the examples of the pairs that improved ϵ. For example, a metonymic expression (I1) was included in a user question, and the relevant text contained its interpretative expression. Due to the matching of these expressions, the score of this text overcame those of irrelevant texts.

Table 3.10 shows all the pairs that worsened ϵ ((W1) worsened ϵ of two user questions). (W1)-(W4) of the pairs are evidently incorrect as interpretations for metonymy. By contrast, although (W5)-(W7) are correct as interpretations for metonymy, they worsened ϵ: they accidentally exposed flaws in other matching methods proposed in Chapter 2. As for (W5), a user question (3.13) was misguidedly matched with an irrelevant text (3.14): underlined parts indicate the expressions of the pair.

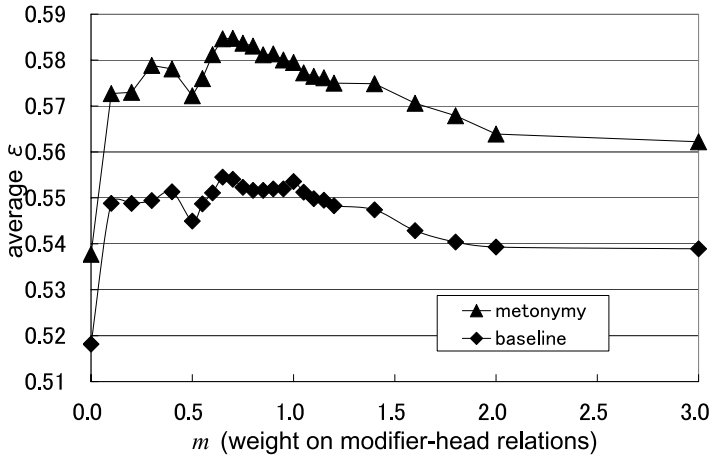
(3.13) *WindowsXP*から *Web*フォルダに アクセスできない
WindowsXP-kara *Web-foruda-ni* *akusesu-deki-nai*
WindowsXP-abl Web folder-acc access-(can)-(not)
'I can not access Web folder from WindowsXP.'

(3.14) *WindowsXP* 環境の *Access*は, ... *UNICODE 0x00A5*を
WindowsXP *kankyō-no* *Access-wa,* ... *UNICODE 0x00A5-wo*
WindowsXP environment-gen Access-TM, ... UNICODE 0x00A5-acc
使用します
shiyō-shimasu
use-(politeness)
'Access on WindowsXP environment uses UNICODE 0x00A5 ...'

The reasons for the mismatch were as follows: the recursive relation between "アクセス" (*akusesu*, access) and *"Access"* (a database application produced by Microsoft Corporation) had been unsuitably expanded (see Subsection 2.4.1); and the matching algorithm for synonymous expressions ignored the difference of the case markers $\boldsymbol{P_\alpha}$="から" (*kara*, ablative case marker) and $\boldsymbol{P_\beta}$="の" (*no*, genitive case marker).



(Help texts: 31 user questions)



(Support KB: 140 user questions)

Figure 3.4: Evaluation of the performance of the metonymic expressions on testsets

Table 3.8: Number of questions on which ϵ was improved or worsened

testset	improved	worsened
Help texts	6 (1)	0 (0)
Support KB	13 (2)	3 (2)

(Numbers in brackets means the cases that the user questions include interpretative expressions, and corresponding metonymic expressions are in matched texts)

3.6 Related Work

As we have pointed out at the beginning of this chapter, processing metonymy is very important for various NLP applications. As for machine translation, Kamei *et al.* [41] indicated that the following sentence (and its literal translation) is fully acceptable in English and Japanese, but it is unacceptable in Chinese:

(3.15) He read Mao.

To make English-Chinese or Japanese-Chinese translation of (3.15), it is inevitable to recognize that (3.15) is a metonymic expression. Furthermore, Harabagiu [25] pointed out that processing metonymy is useful for anaphora resolution. In the following passage, recognizing that "The White House" is used to refer "administration" validates the coreference link.

(3.16) The White House sent its health case proposal to the Congress yesterday.
Senator Dole said the administration's bill had little chance of passing.

Most of previous studies on recognition and processing of metonymy were based on manually constructed ontologies, semantic frames, or logical forms. Fass [42] proposed the *met** method, which recognizes semantic relations (*e.g.* metonymy, metaphor, and so on) between pairs of word senses in given sentences, based on some rules for the relations and sense-frames for verbs and nouns. As for (3.17), the method judges that it is not literal, by detecting violation of contextual constraints based on two sense-frames for *play* and *Bach*. After that, it tests the relation between *play* and *Bach* with each rule, and the rule for *ARTIST FOR ART FORM* is satisfied.

(3.17) Ted played *Bach*.

Table 3.9: Pairs which improved ϵ

	metonymic expressions			interpretative expressions			
	A_α	P_α	V_α	A_β	B_β	P_β	V_β
(I1)	[user question]			[Help text]			
	LAN	で	接続	*LAN*	経由	で	接続
	LAN	*de*	*setsuzoku*	*LAN*	*keiyu*	*de*	*setsuzoku*
	LAN	ins	connect	LAN	through	ins	connect
	'connect by LAN'			'connect through LAN'			
(I2)	[user question]			[Help text]			
	ファイル	に	関連づける	ファイル	の 種類	に	関連づける
	fairu	*ni*	*kanrendukeru*	*fairu*	*shurui*	*ni*	*kanrendukeru*
	file	dat	associate	file	type	dat	associate
	'associate with files'			'associate with a file type'			
(I3)	[user question]			[Help text]			
	文字	を	大きくする	文字	の サイズ	を	大きくする
	moji	*wo*	*ookiku-suru*	*moji*	*saizu*	*wo*	*ookiku-suru*
	character	acc	enlarge	character	size	acc	enlarge
	'enlarge characters'			'enlarge character size'			
(I4)	[user question]			[Support KB]			
	HTML	で	保存	*HTML*	形式	で	保存
	HTML	*de*	*hozon*	*HTML*	*keishiki*	*de*	*hozon*
	HTML	ins	save	HTML	format	ins	save
	'save as an HTML'			'save in HTML format'			
(I5)	[user question]			[Support KB]			
	画像	を	開く	画像	ファイル	を	開く
	gazō	*wo*	*hiraku*	*gazō*	*fairu*	*wo*	*hiraku*
	gazō	acc	open	image	file	acc	open
	'open an image'			'open an image file'			
(I6)	[user question]			[Support KB]			
	ユーザー	を	登録	ユーザー	情報	を	登録
	yūzā	*wo*	*tōroku*	*yūzā*	*jōhō*	*wo*	*tōroku*
	user	acc	register	user	information	acc	register
	'register a user'			'register user information'			
(I7)	[user question]			[Support KB]			
	名刺	を	作成	名刺	の デザイン	を	作成
	meishi	*wo*	*sakusei*	*meishi*	*dezain*	*wo*	*sakusei*
	visiting card	acc	make	visiting card	design	acc	make
	'make visiting cards'			'make visiting card design'			
(I8)	[user question]			[Support KB]			
	システム	が	不安定だ	システム	の 動作	が	不安定だ
	shisutemu	*ga*	*fuantei-da*	*shisutemu*	*dōsa*	*ga*	*fuantei-da*
	system	nom	unstable	system	operation	nom	unstable
	'a system is unstable'			'system operation is unstable'			
(I9)	[Support KB]			[user question]			
	アプリケーション	が	遅い	アプリケーション	の 起動	が	遅い
	apurikēshon	*ga*	*osoi*	*apurikēshon*	*kidō*	*ga*	*osoi*
	application	nom	slow	application	launch	nom	slow
	'an application is slow'			'a launch of an application is slow'			
(I10)	[Support KB]			[user question]			
	履歴	を	削除	履歴	の 情報	を	削除
	rireki	*wo*	*sakujo*	*rireki*	*jōhō*	*wo*	*sakujo*
	history	acc	delete	history	information	acc	delete
	'delete the history'			'delete the history information'			

Table 3.10: Pairs which worsened ϵ

	metonymic expressions			interpretative expressions			
	A_α	P_α	V_α	A_β	B_β	P_β	V_β
(W1)	[user question]			[Support KB]			
	ページ	を	表示	ページ	の 番号	を	表示
	pēji	*wo*	*hyōji*	*pēji*	*bangō*	*wo*	*hyōji*
	page	acc	display	page	number	acc	display
	'display a page'			'display page numbers'			
(W2)	[user question]			[Support KB]			
	DNS	を	使う	*DNS*	の 動的更新	を	使う
	DNS	*wo*	*tsukau*	*DNS*	*dōteki-kōshin*	*wo*	*tsukau*
	DNS	acc	use	DNS	dynamic update	acc	use
	'use DNS'			'use DNS dynamic update'			
(W3)	[user question]			[Support KB]			
	アプリケーション	を	開く	アプリケーション	の ファイル	を	開く
	apurikēshon	*wo*	*hiraku*	*apurikēshon*	*fairu*	*wo*	*hiraku*
	application	acc	open	application	file	acc	open
	'open an application'			'open an application file'			
(W4)	[Support KB]			[user question]			
	ファイル	を	印刷	ファイル	一覧	を	印刷
	fairu	*wo*	*insatsu*	*fairu*	*risuto*	*wo*	*insatsu*
	file	acc	print out	file	list	acc	print out
	'print out a file'			'print out a list of files'			
(W5)	[user question]			[Support KB]			
	Windows XP	から	アクセス	*Windows XP*	環境	から	アクセス
	Windows XP	*kara*	*akusesu*	*Windows XP*	*kankyō*	*kara*	*akusesu*
	Windows XP	abl	access	Windows XP	environment	abl	access
	'access from Windows XP'			'access from Windows XP environment'			
(W6)	[user question]			[Support KB]			
	文字	を	囲む	文字	列	を	囲む
	moji	*wo*	*kakomu*	*moji*	*retsu*	*wo*	*kakomu*
	character	acc	circle	character	string	acc	circle
	'circle characters'			'circle a character string'			
(W7)	[Support KB]			[user question]			
	XP	で	送信	*XP*	の パソコン	で	送信
	XP	*de*	*sōshin*	*XP*	*pasokon*	*de*	*sōshin*
	XP	ins	submit	XP	PC	ins	submit
	'submit ... by (Windows) XP'			'submit ... by (Windows) XP PC'			

Stallard [43] proposed a method based on logical forms, for distinction between two kinds of metonymy: "referential" metonymy, in which the referent of a nominal predicate argument requires coercion (*e.g.* (3.18)), and "predicable" metonymy, featuring the coercion of the predicate usually corresponding to a verbal lexicalization (*e.g.* (3.19)).

(3.18) Which wide-body jets serve dinner?

(3.19) Which airlines fly from Boston to Denver?

Murata *et al.* [44] proposed a method based on manually-constructed case frames [38]. As for (3.20), the case frame for "読む" (*yomu*, read) is looked up, and "トルストイ" (*torusutoi*, Tolstoy) is recognized as a metonymic word, because it does not satisfy the semantic restriction by the case frame. After that, examples in the forms of "Noun ***A*** *(の; no)* Noun ***B***" are gathered from a corpus. As a result, the examples such as (3.21) are gathered, and "小説" (*shōsetsu*, novel) is found as the interpretation.

(3.20) 僕が トルストイを 読む.
boku-ga *torusutoi-wo* *yomu.*
I-nom Tolstoy (a novelist)-acc read
'I read Tolstoy.'

(3.21) トルストイの 小説
torusutoi-no *shōsetsu*
Tolstoy-gen novel
'Tolstoy's novel'

Some methods were based on lexical database. Harabagiu [25] proposed a method for deriving metonymic coercions based on WordNet [36] and operations of logical forms. As for (3.16), the method detects the co-reference between "The White House" and "administration" using relations in WordNet. Peters [48] explored sets of concepts comprising possible metonymic relations in EuroWordNet [49].

However, the above methods have a critical problem: their knowledge structures require heavy cost of construction and maintenance, and makes scaling up quite difficult. Therefore, recent studies on processing metonymy adopted corpus-based approaches.

Utiyama *et al.* [45] proposed a method for giving interpretations for inputted metonymies in the form of "Noun ***A*** Case-Marker ***R*** Predicate ***V***", using statistical measures on a large corpus. As for (3.22), "湯" (*yu*, hot water), "お湯" (*oyu*, hot water), and "熱湯" (*nettō*, boiling water) are given as the interpretations.

(3.22) やかんが　沸く
yakan-ga　*waku*
kettle-nom　boil
'the kettle is boiling'

Also, Lapata *et al.* [46] proposed a method for interpreting *logical metonymy* based on a probabilistic model on a large corpus. For example, it gives "tell" for (3.23), and "solve" for (3.24) as their interpretations.

(3.23) John began the story.

(3.24) easy problem

Our study has some advantage over these previous studies. First, this study revealed a critical problem caused by metonymy with full-parsing-based matching methods, which are generally important for information retrieval and question-answering. Second, this study proposed a method which can be applied to matching metonymic expressions on real-world applications, and demonstrated that the method was effective on a concrete application. In addition, this study treated both recognition and interpretation process of metonymy. The type of metonymy this study handled was very similar to those of Utiyama *et al.* [45] and Lapata *et al.* [46]; however, their works targeted only on the interpretation process of metonymy, and left the recognition process for a future work.

3.7 Summary of this Chapter

This chapter proposed a method of acquiring metonymic expressions and their interpretative expressions from the corpora, including a lot of questions of novice users collected by our system. Furthermore, we applied the acquired metonymic expressions into matching user questions with texts, and showed the effectiveness of our method.

Our method also extracts some wrong interpretations of metonymy, and sometimes they worsen the performance. Moreover, our method treated only a part of the metonymic world. To cope with such problem, we have to study a synthetic model of metonymy.

Chapter 4

User Navigation

4.1 Introduction

Chapter 2 and Chapter 3 have described the methods for matching a user question with relevant texts, based on several NLP techniques, including sentence structure analysis. However, even if a user ordinarily says what he/she wants to know, it is rare that one relevant text is unambiguously determined. Consider a user question "An error has occurred while booting Windows 98": this question is relatively specific, however, there are multiple causes and solutions for his/her problem, and each cause and solution has a text. If a user question is even vaguer, much more texts are selected. Anyway, a user has to select one best text that fits his/her situation, from multiple texts as candidates.

We preliminarily examined the question logs for the natural language based text retrieval system *Hanashi-kotoba Kensaku*[1] serviced by Microsoft Japan, and categorized questions. The examination shows that about 30% of the user questions are vague. The application of proposed matching methods for such vague questions without any supports will result in a pile of matched texts. If all the texts are shown at once, it is hard for a user to find a relevant one for the situation he/she is coming up with. To cope with the problem, asking-backs for clarifying vague questions are required.

Some web search engines help a user's selection of a text, by showing the part that contains the query keywords, as a context. Advancing this methodology, based on the flexible and precise matching methods described in Chapter 2, we proposes a novel method of making asking-backs, extracting descriptions that clarify situations of user questions (encountered problems), and making asking-backs by showing such descriptions as candi-

[1] http://www.microsoft.com/japan/enable/nlsearch/

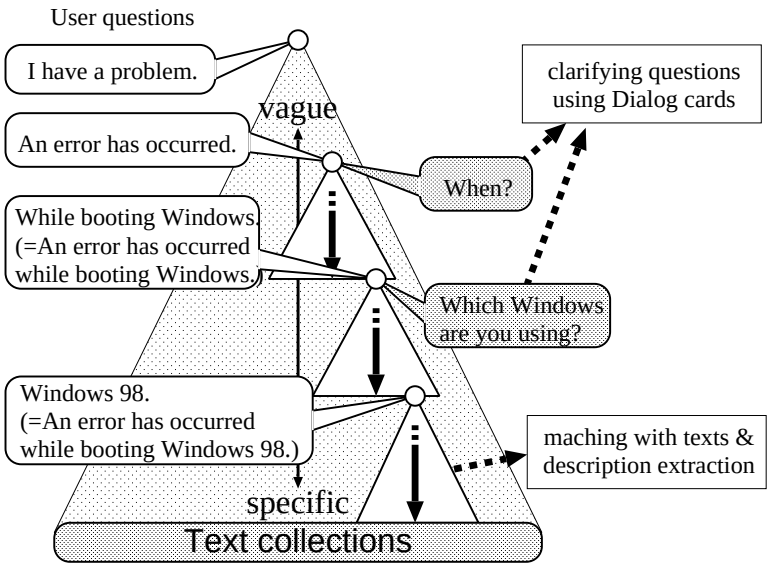


Figure 4.1: User navigation

dates.

However, if a user question is very vague, the above method often fails, because it is too difficult to extract such descriptions. Moreover, such a vague question will match too many texts, so the user will have to pay his/her labor on finding a relevant text. Therefore, we systematized procedures for making asking-backs to clarify frequently asked vague questions, that is, *Dialog cards*. Using the cards, a user's problem will be specified through a dialog, with some asking-backs. Note that the matching methods described in Chapter 2 are also applied for matching a user question with the dialog cards.

Figure 4.1 shows a strategy for navigating a user from vague questions to specific texts efficiently, by using the above two methodologies complementarily. The biggest triangle in the figure means the collection of user questions which have a variety of vagueness: the upper part of the triangle corresponds to vaguer questions; the lower part of the triangle corresponds to more specific questions; and the text collections place at the bottom of the triangle, because every text is very specific. For slightly vague questions such as "An error has occurred while booting Window 98", the question is matched with texts by our proposed methods described in Chapter 2 and Chapter 3, and the description extraction method is applied for each matched text. By contrast, for vaguer questions such as "An error has occurred", those questions are clarified step-by-step, by making

asking backs such as "When?" and "Which Windows are you using?", and then the description extraction method is also applied.

In other words, our strategy for clarifying vague questions consists of two approaches: a top-down approach as dialog cards, and a bottom-up approach as description extraction.

This chapter describes the above two methods in the following two sections: Section 4.2 gives the method of description extraction, and Section 4.3 gives the method for making asking-backs using *Dialog cards*. And then, Section 4.4 compares our methods with those of previous work. Finally, Section 4.5 concludes this chapter.

4.2 Asking-Backs by Description Extraction

In most cases, the neighborhood of the part that matches a user question describes specific symptoms and conditions of the problems users often encounter. Note that the word *neighborhood* is used to indicate a part that places next to the matching part, but that does not match with a user question itself. For example, when a user question "A page fault occurs" is matched with a text sentence "A page fault occurs while launching IE5", the neighborhood part "while launching IE5" specifies the user's situation. Extraction of such neighborhood from each text that matches a user question will support a user's selection, especially if several texts are matched with the question. We define the neighborhood as *description*, and such extraction as *description extraction*.

Our method of description extraction is applied to each sentence that matches a user question as follows:

1. Remove frequently occurred verbose expressions such as "この資料では..." (*kono shiryō-dewa ...*, in this article, ...), "... 問題について説明しています" (*... mondai-ni tsuite setsumei-shite-imasu*, describes the problem that ...), and "以下の..." (*ika-no ...*, the following ...) by pattern matches.

2. Segment each sentence into a few parts (*segments*) based on some linguistic criteria. Our method considers the following points as a boundary for segmentation:

   - After a verb indicating a subordinate clause.

     Japanese is a head-final language, and arguments of each verb are placed left to the verb. If the conjugation of a verb indicates that it modifies a verb, it means that the verb is a head of a subordinate clause.

- After a noun indicating a conditional clause.

  “とき” (*toki*, in time of), “際” (*sai*, in case of), “場合” (*baai*, in case of) and “最中” (*saichū*, in the course of) are regarded as such nouns.

- After a postposition “で” (*de*) with a comma.

  The postposition “で” (*de*) is an ambiguous case marker indicating place, method or reason. However, whatever it indicates, the case components of place/method/reason have a relatively weak connection to the main verb.

- Coordinations are separated into conjuncts.

3. Remove segments of which every *bunsetsu* has correspondence with that of a user question (see Subsection 2.7.1).

4. Select the last segment (excluding deleted segments) as a nucleus of the description.

5. Select the nucleus itself and segments that directly modify the nucleus as the description.

Figure 4.2 shows an application of the above algorithm for two sentences as answer candidates. First, the left sentence is divided into two segments **A** · **B**, and the right sentence is divided into three segments **C** · **D** · **E**. Next, **B** in the left sentence and **C** · **E** in the right sentence are removed, because every *bunsetsu* of them has correspondence with the user question. As a result, **A** and **D** each is selected as a nucleus of a description. Finally, “*IE5*を起動した際に” (*IE5-wo kidō-shita sai-ni*, while launching IE5) and “タスクスケジューラを使うと” (*tasukusukejūra-wo tsukau-to*, when the task scheduler is used) are outputted as descriptions of matched texts.

The above method is applied only to representative sentences (see Subsection 2.7.2) of texts of Support KB. As to Glossary and Help texts, the method is not applied, because their entries or titles themselves are brief descriptions.

4.3 Asking-Backs by Dialog Cards

If the user question is too vague, it is too difficult to detect matching parts with the user question, and then the description extraction often fails. Moreover, such a question matches many texts, so users have to pay their labor on finding a relevant one.

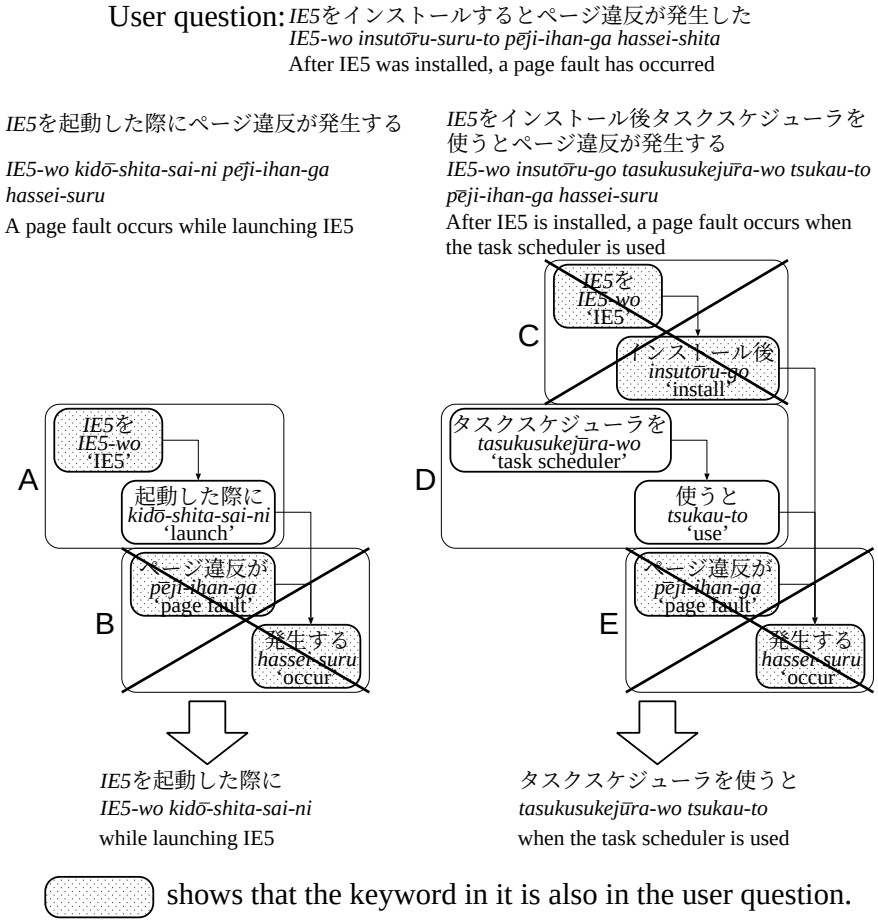


Figure 4.2: Description extraction from matched text sentences

<CARD>
<ID> [Error]
<UQ> エラーが発生する ‘An error is caused’
<REPLY> エラーはいつ発生しますか? ‘When is the error caused?’
<SEL action=CARD card_id=[Error/Booting Windows]>
Windows 起動中 ‘while booting Windows’
<SEL action=CARD card_id=[Error/Printing Out]>
印刷時 ‘while printing out’
<SEL action=CARD card_id=[Error/Launching Applications]>
アプリケーション起動時 ‘while launching applications’
</CARD>

<CARD>
<ID> [Error/Booting Windows]
<UQ> *Windows* を起動時にエラーが発生する ‘An error is caused while booting Windows’
<REPLY> あなたがお使いの *Windows* を選んでください. ‘Which Windows are you using?’
<SEL action=RET phrase=*Windows 95* を起動時にエラーが発生する ‘An error is caused while booting Windows 95’>
Windows 95
<SEL action=RET phrase=*Windows 98* を起動時にエラーが発生する ‘An error is caused while booting Windows 98’>
Windows 98
<SEL action=RET phrase=*Windows ME* を起動時にエラーが発生する ‘An error is caused while booting Windows ME’>
Windows ME
</CARD>

Figure 4.3: Dialog cards

We therefore systematized procedures for interactively clarifying frequently asked vague questions as *dialog cards*. The cards were constructed based on the manuals for the operators of the call center in Microsoft Corporation. Figure 4.3 shows two examples of dialog cards. A dialog card describes how to make asking-backs for a user question, including the following elements:

<ID>: An ID of the dialog card.

<UQ>: If this part is matched with a user question, this dialog card is used for making

U: エラーになった ‘An error was caused.’
S: エラーはいつ発生しますか? ‘When was the error caused’
1. *Windows* 起動中 ‘while booting Windows’
2. 印刷時 ‘while printing out’
3. ...
U: *Windows* 起動中 ‘while booting Windows’
S: あなたがお使いの *Windows* を選んでください. ‘Which Windows are you using?’
1. *Windows 95*
2. *Windows 98*
3. ...
U: *Windows 98*
S: 以下の選択肢から選んでください. ‘Please select the following choices.’
1. *W98:*起動時のフォントドライバが読み込めないエラー ‘W98: An error in trying to load font drivers on start-up’
「*JIS* フォントドライバーがインストールされていません」等のフォントの読み込みに関するエラーが発生した ‘Font loading errors such as “The JIS font driver has not been installed” were caused’
2. *W98:*起動時に「< ファイル名 > が不正かありません」のエラーについて ‘W98: About an error “[filename] is illegal or not found” on start-up’
3. ...

Figure 4.4: A dialog using dialog cards

an asking-back.

<REPLY>: An asking-back for a user.

<SEL action=CARD/SHOW/RET ... >: A Choice with the asking-back. Each choice describes the next action that a system has to do when a user selects this choice: action=CARD means that the dialog card indicated by card_id should be used next; action=SHOW means that a web page on the Microsoft's site indicated by url or a text of the text collections indicated by text_id should be displayed; and action=RET means that the matching of a question indicated by phrase with texts should be done.

Our method uses the dialog cards based on the matching of a user question with <UQ> parts, using the method described in Chapter 2: if dialog cards are matched at high scores (≥ 0.8), the card that has the highest score is used for making an asking-back (see Subsection 2.7.4 and Table 2.4).

Figure 4.4 shows an dialog using the two dialog cards in Figure 4.3. First, a matching user question “エラーになった” (*erā-ni natta*, An error was caused) with each <UQ> part

of every dialog card by the method described in Chapter 2 selected the above dialog card in Figure 4.3. Next, following the dialog card, the first asking-back “エラーはいつ発生しますか?” (*erā-wa itsu hassei-shimasu-ka?*, When was the error caused?) was made with some choices. After the user selected “*Windows*起動中” (*Windows kidō-chū*, while booting Windows), following the below dialog card in Figure 4.3 as indicated by this choice, the another asking-back “あなたがお使いの *Windows*を選んでください” (*anata-ga otsukai-no Windows-wo erande-kudasai*, Which Windows are you using?) was made. Finally, after the user selected *“Windows 98”*, as indicated by this choice, a matching of the query “*Windows 98*を起動時にエラーが発生する” (*Windows 98-wo kidō-ji-ni erā-ga hassei-suru*, An error is caused while booting Windows 98) with the text collections were made, and the result of the matching was showed.

As the example shows, the dialog cards have a hierarchical structure, and the <UQ> of every dialog card is targeted of the matching with each user question: that is, the structure of the dialog cards was designed to cover various user questions which have a variety of vagueness as shown in Figure 4.1. For example, if a user asked a question “*Windows*を起動時にエラーが発生する” (*Windows 98-wo kidō-ji-ni erā-ga hassei-suru*, An error is caused while booting Windows), the asking-back using the bottom dialog card in Figure 4.3 is made first.

In addition, the framework of dialog cards is also applied for making exceptional responses which are out of the target domain, such as U:“こんにちわ” (*konnichiwa*, Hello) S:“こんにちは” (*konnichiwa*, Hello) and U:“このシステム使いやすいですね” (*kono shisutemu tsukai-yasui-desu-ne*, This system is easy to use) S:“ありがとうございます” (*arigatō-gozai-masu*, Thank you). In such cases, dialog cards that have no <SEL> are used. Such responses are very important for keeping proper dialogs: without such responses, a matching of a user question “このシステム使いやすいですね” (*kono shisutemu tsukai-yasui-desu-ne*, This system is easy to use) would result in showing irrelevant texts that have general keywords such as “システム” (*shisutemu*, system) and “使う” (*tsukau*, use). In the next chapter, we evaluate such exceptional dialogs as out-of-domain ones.

4.4 Related Work

As we have mentioned in Subsection 1.2.3, efforts on realizing some sort of interactions with users have been done from the following two approaches: extensions of text re-

trieval systems, and application of artificial intelligence techniques. This section gives an overview of these former studies and some other related work, and compared them with our approach (Table 4.1).

Although basic keyword-based text retrieval systems have no capability of making such asking backs, some studies have tried to implement the capability to the systems. These studies involve the following methods:

- Asking-backs by texts.

  A lot of previous systems including SMART [18] were based on a method that requests a user to select relevant texts, and feedbacks the result into modification of a query[2]. This method is also adopted some contemporary web search engines such as Google[3]: each retrieved text has a button "similar pages", that seems a method for asking a user for a relevant text.

- Asking-backs by related keywords.

  Several systems such as RCAAU [50], DualNAVI [51], and Excite [4] use a method that shows keywords related to a query, and request a user to select one (some) of them.

- Asking-backs by both texts and related keywords.

  THOMAS [52] uses the following method: a user's query is kept as an *image*, that is, a collection of keywords; the system gradually modifies the *image*, by repeating a process that requests the user to select one relevant text and related texts. However, the system was proposed in 1970s, and it is applicable only to small text collections.

- Clustering.

  Some systems including Scatter/Gather [19] and WEBSOM [20] use a method that shows clusters as choices, based on automatic classification algorithms. The clusters are usually represented by texts that belong to the cluster, or lists of representative keywords.

[2]Ordinarily, this method is called as *relevance feedback*. However, taking this word in a wide sense, it indicates the whole methods that request a user some information. Therefore, we do not use the word here.

[3]`http://www.google.com/`

[4]`http://www.excite.com/`

Table 4.1: Various types of information retrieval systems

methods or systems	user questions	outputs	asking-backs	scales
basic keyword-based text retrieval systems	a collection of keywords	a collection of texts	N/A	large
asking-backs using texts (SMART, web search engines)	a collection of keywords	a collections of texts	texts	large
asking-backs by keywords (RCAAU, DualNAVI, Excite)	a collection of keywords	a collection of texts	related keywords	large
asking-backs by both texts and keywords (THOMAS)	collections of keywords	a text	texts and keywords	small
clustering (Scatter/Gather, WebSOM)	a collection of keywords	a collection of texts	clusters represented by texts or keywords	large
exploitation of a body of knowledge described in formal languages (UC)	natural languages	natural languages (answers)	natural languages	very small
exploitation of FAQ texts (FAQ Finder)	natural languages	natural languages (answers)	N/A	small
exploitation of open-domain texts (TREC QA/NTCIR QAC)	natural languages	natural languages (answers)	N/A	large
Dialog Help system of CIMS, Kyoto University	natural languages	natural languages (answers)	natural languages	small
our approach	natural languages	natural languages (descriptions)	natural languages	large

The above methods use texts or keywords as media for making asking-backs. However, these media are not always suitable because of their limitations: keywords have less power of expression, because they are too abstract; in contrast, texts are too specific.

By contrast, some systems based on the artificial intelligence techniques in 1980s had the capability of making appropriate asking-backs. For example, UC (Unix Consultant) [21] was equipped with an ability of making asking-backs using natural languages if user questions were not specific. However, these systems were not applicable to existing large text collections, because they required specialized knowledge base described in formal languages. It was too difficult to design an almighty formal language, and construction and maintenance of such knowledge base required heavy cost.

Meanwhile, as large electronic text collections became available from 1990s, QA systems that exploit such collections as knowledge base have been studied. Generally, they adopt deeper NLP techniques including sentence structure analysis based on full parsing in order to find exact answers. For example, FAQ Finder [8] exploits a lot of FAQ texts on the USENET, and applies case analysis on the texts. Recently, open-domain question answering systems that exploits unstructured texts such as newspapers have been actively studied in TREC QA Track [5] or NTCIR QAC [9] ([12–14, 34]). These systems also adopted sentence structure analysis and question type analysis for extracting exact answers. However, the above systems have no capability of making asking-backs: they only reply an answer for each user question, based on the assumption that every user question is specific.

Dialog Help System of CIMS at Kyoto University [26] is a system that has capabilities of making asking-backs using natural language, based on a flexible matching of a user question with knowledge base described in natural language. However, it requires knowledge base described in uniform and limited expressions, so it seems a prototype system for *asking-backs for vague questions*.

In contrast, our approach realizes *asking-backs for vague questions*, based on real-world large text collections. The major novelty of our approach is derived from the method of description extraction. By proposing the method, we showed that the methodology of QA systems, *i.e.,* extraction of exact answers based on full parsing, is also applicable to making asking-backs for vague questions. Although the description extraction may result in insufficient asking-backs especially in case of so many matched texts, we think that the method will be an important basis of making more sophisticated asking-backs.

Another method of our approach, asking-backs by dialog cards, also has some originality. Unlike the methods based on artificial intelligence techniques, the dialog cards describe knowledge structures of making appropriate asking-backs like experts *by natural languages*. As a result of using natural languages, the dialog cards have several advantages: they work reasonably without rigid descriptions; user questions at various levels of vagueness are subject to be asked by the dialog cards which have a hierarchical structure; a variety of expressions in user questions are resolved by the flexible matching methods described in Chapter 2 and Chapter 3.

4.5 Summary of this Chapter

At the beginning of this chapter, we pointed out the importance of making asking-backs for vague questions for real-world information retrieval systems. After that, we proposed the strategy for realizing such asking-backs by combining the two approaches: *description extraction* as the bottom-up approach and *dialog cards* as the top-down approach. Both approaches are based on the precise and flexible matching methods described in Chapter 2. In addition, we discussed the difference between our method and those of previous studies on making asking-backs, and showed the novelties of our method.

The next chapter will propose a specific architecture to realize the proposed method, and will show results on a real-world operation.

Chapter 5

Dialog Navigator

5.1 Introduction

Most of previous studies on information retrieval systems mainly evaluated the systems by *precision* and *recall* on test collections. However, test collections are not available for evaluating systems that involve interactions with users. To evaluate such systems, task-oriented evaluations should be done.

The task-oriented evaluations were mainly based on a strategy that showed subjects some scenarios and made them ask questions in accordance with the scenarios. However, to evaluate our method of making asking-backs for vague user questions in line with the discussions in Chapter 1, the strategy is unsatisfactory. It is too difficult to determine collections of scenarios that reflect situations on the real world, including vague questions about troubles with industrial products. Once a scenario is given, a subject's *artificial* question inevitably becomes specific, without vagueness. To deal with the dilemma, a real-world evaluation is required. That is, the system should be evaluated by motivated users who have troubles on using products and seek for solutions.

To achieve such real-world evaluation, we implemented the proposed methods on a dialog system based on the real-world text collections provided by Microsoft Corporation: *Dialog Navigator*, which targets ordinary users who use Windows operating systems on personal computers. And then, we got a lot of ordinary Windows users to use the system, and evaluated the system based on the real-world operation results, that is, the stored logs of user questions and system responses.

Dialog Navigator started its service on the web site of Microsoft Corporation (`http://www.microsoft.com/japan/navigator/`) in April 2002, and all conversation logs of user questions and system responses have been stored as a *dialog database*.

Figure 5.1: User interface for *Dialog Navigator*

This chapter describes the real-world operation of our methods proposed in the previous chapters. First, Section 5.2 gives an interface of Dialog Navigator, which enables users to make efficient dialogs. Next, Section 5.3 shows a specific architecture for realizing the method described in Chapter 4, which makes asking-backs for vague user questions. And then, Section 5.4 shows evaluations of the dialog database of Dialog Navigator as the effectiveness of our proposed methods, from a couple of aspects. Finally, Section 5.5 concludes this chapter.

5.2 User Interface

To get a lot of user accesses, we adopted a web-based interface for *Dialog Navigator*. Users can easily access the system by common web browsers, without installing any additional software in personal computers. As Figure 5.1 showed, the user interface of Dialog Navigator has two frames: the upper frame shows dialog histories between a user and the system, with a text box for inputting user questions by a keyboard; the lower frame shows some choices when the system makes an asking-back, and a user selects one of the

choices by a mouse.

The system uses several devices to improve the usability:

- *Size of the text box:*

  The text box for inputting user question has larger size than usual web search engines. It induces users to input not keywords, but sentences.

- *Icons with matched texts:*

  When the system shows matched texts as choices, each text is displayed with a colored bar graph, which indicates the matching score by its length and the type of the text by its color (red means Glossary, yellow means Help texts, and blue means Support KB). It enables users to know the information about each text intuitively.

- *Ordering of matched texts:*

  Matched texts are shown in order of easiness, that is, Glossary, Help texts, and Support KB. Texts of Support KB are especially difficult for novice users, so the order reduces confusion for such users.

- *Special treatment for what type questions:*

  If a question type estimation of a user question (see Subsection 2.3.4) is *what* type, and the question is matched to one entry of Glossary with the maximum score (= 1.0), the definition is directly displayed in the upper frame.

- *A pop-up window for displaying texts:*

  When one of matched texts is clicked, the content of the text is displayed on a pop-up window.

5.3 Architecture

Figure 5.2 shows the flow chart of *Dialog Navigator*, including internal processes of the system and user actions. Basically, the following three steps are taken in order:

1. The user question is clarified through a dialog with the user based on dialog cards (the left side loop of Figure 5.2).

2. The clarified user question is matched with the text collections (moving to the right side of Figure 5.2).

3. Descriptions of the matched texts are extracted automatically, and those are shown to the user as choices.

Note that if the user question is specific, the matching result with text collections is directly displayed, skipping the dialog processes with dialog cards.

Each component in Figure 5.2 works as follows:

- *Input Analyzer:*

  It applies the sentence structure analysis methods described in Section 2.3 to each user question. As a result, each user question is parsed into a dependency structure between *bunsetsu*, keywords are extracted from, assigned negation flags and one of question types (*i.e. what* type, *how* type, *symptom* type, and *no* type), and verbose final expressions are removed from. In addition, synonymous expressions are extracted from each question (see Subsection 2.4.1).

- *Text Matcher:*

  Based on the method described in Section 2.6 and Section 2.7, it matches the user question with the text collections or the dialog cards, and returns texts (or cards) that have high scores. Expression gaps between the user question and the texts (or cards) are resolved by the synonymous expression dictionary and the ontological dictionary (see Section 2.4).

- *Description Extractor:*

  Based on the method described in Section 4.2, it shows brief choices to the user, by extracting neighborhoods of the part that matches the user question, from each matched text.

5.4 Evaluation and Discussion

To evaluate the proposed methods on *Dialog Navigator*, we randomly selected some dialogs from the dialog database, and examined them from the following three aspects:

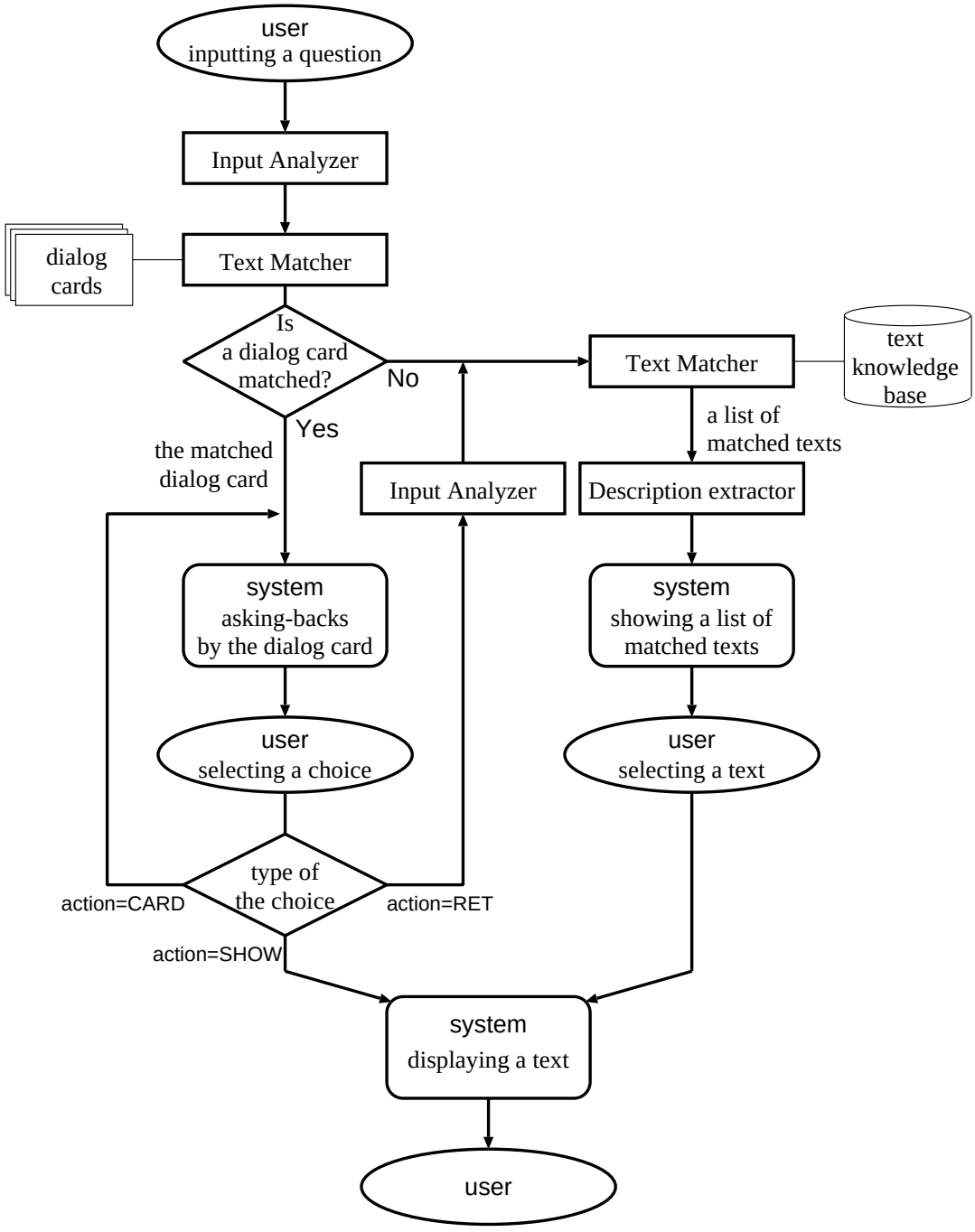


Figure 5.2: The flow chart of *Dialog Navigator*

- *Does the system return relevant answers?* (Subsection 5.4.1)

  We divided the dialogs into session units, and a subject judged whether each session has relevant matched texts, based on estimation of user intentions. This subsection defines detailed criteria for the examination, and compares the results with dialog cards to those without dialog cards.

- *How do users and the system behave in each dialog?* (Subsection 5.4.2)

  We validated the strategy for making asking-backs, based on frequency distribution of behaviors of users and the system, and relations between length of user questions and system responses. In addition, we checked whether numbers of choices were appropriate for users or not.

- *How do the results of description extraction help users?* (Subsection 5.4.3)

  A subject evaluated the effectiveness of description extraction, based on some criteria.

5.4.1 Evaluation of Dialog Sessions

As the target for the examination, we randomly selected 300 accesses from a part of the dialog database: 1st - 31st August 2002. Next, we manually segmented each accesses into task-oriented units (we call the units a *dialog session*). As a result, we got 378 dialog sessions. On average, one dialog session had 1.42 utterance pairs (each utterance pair means a combination of a user question input and a system response). The average length of user questions was 2.5 *bunsetsu*: it showed that the users usually inputted questions rather as sentences than as keywords. We think that the users were induced to input longer questions by the larger size of the text box.

For each dialog session, a subject gave either of the following evaluations, by estimating the user's intention:

- **success:text**

  The system showed at least one relevant text as choices.

- **success:no_text**

  The system had no relevant texts, and answered that no relevant texts were found. We regard this evaluation as successful one, because it is appropriate for the system to answer so.

Table 5.1: Evaluations of dialog sessions (session type A(1))

A-1	U:	Excelで行を追加したい 'I want to add columns in Excel'
	S:	以下の選択肢から選んでください．'Please select the following choices.'
		1. ○ Excelで空白のセル、行、列を挿入する'Inserting empty cells, columns, or rows in Excel'
		2. ○ EXCEL: 行の挿入 'EXCEL: insertion of columns'
		3. ...
	evaluation	**success:text**
A-2	U:	起動ディスク 'boot disk'
	S:	以下の選択肢から選んでください．'Please select the following choices.'
		1. W98: PC-9800シリーズ対応用起動ディスクの起動メニュー 'W98: boot menu of the boot disk for PC-9800 series'
		2. Windows XP インストール時にフロッピーでコンピューターを起動するためのディスクの入手方法 'How to get a floppy disk to boot a computer for installing Windows XP'
		3. ...
	U:	Windows 2000での起動ディスクの作成方法は? 'How to make a boot disk on Windows 2000?'
	S:	以下の選択肢から選んでください．'Please select the following choices.'
		1. ○ Windows NTブートディスクを作成しブート障害に備える方法 'How to use a Windows boot disk to prevent boot failure in Windows NT'
		2. Windows 2000 Server の新規インストールを実行する方法 'Perform a clean installation in Windows 2000 Server' CD-ROMドライブからコンピュータを起動できない場合，Windows 2000ブートディスクを作成します． 'If your computer cannot boot from the CD-ROM drive, create Windows 2000 boot disks.'
		3. ○ ...
	evaluation	**success:text**

- **failure:text**

  Although the system had relevant texts, it could not show any of the texts.

- **failure:no_text**

  Although the system had no relevant texts, it showed other irrelevant texts.

- **misc:proper**

  The user question was out-of-domain, and the system replied to the question properly.

- **misc:improper**

  The user question was out-of-domain, and the system replied to the question improperly.

Table 5.2: Evaluations of dialog sessions (session type A(2))

A-3	U:	USB1.1 から USB2.0 に交換できますか? 'Can I replace USB1.1 with USB2.0?'
	S:	入力された質問から，該当する情報を見つけることができませんでした． 'No relevant information was found for your question.'
	evaluation	**success:no_text**
A-4	U:	マクロを使いたい 'I want to use macros'
	S:	以下の選択肢から選んでください． 'Please select the following choices.'
		1. Access でマクロを使ってプロパティを設定する 'Setting properties using macros in Access'
		2. Access Visual Basic でマクロを使ったプロパティの設定 'Setting properties using macros in Access Visual Basic'
		3. ...
	U:	Excel です 'I use Excel'
	S:	以下の選択肢から選んでください． 'Please select the following choices.'
		1. Excel 2000 の起動時に特定のファイルを開く方法 'How to open a particular on launching Excel 2000'
		2. Excel: Web 上に Excel データを保存する方法'Excel: How to save Excel data on the Web'
		3. ...
	proper text	Excel のマクロについて 'about Excel macros'
	evaluation	**failure:text**
A-5	U:	ウインドウズ XP を削除するにはどうしますか? 'How to remove Windows XP?'
	S:	入力された質問から，該当する情報を見つけることができませんでした． 'No relevant information was found for your question.'
	proper text	Windows XP をアンインストールする方法 'How to uninstall Windows XP'
	evaluation	**failure:text**
A-6	U:	ラリースポーツ チャレンジで正しく画面が表示されない 'The screen is not displayed properly in Rally Sports Challenge'
	S:	以下の選択肢から選んでください． 'Please select the following choices.'
		1. W98SE: 初期設定の画面で [ESC] を連打すると表示が不正 'W98SE: the initial setting screen is displayed incorrectly if [ESC] is typed repeatedly'
		2. [CHT2] キャラクターが正しく表示されない場合 '[CHT2] in case that characters are not displayed correctly'
		3. ...
	evaluation	**failure:no_text**

Table 5.3: Evaluations of dialog sessions (session type B)

B-1	U:	表を書きたい 'I want to draw a table'
	S:	入力された質問から，該当する情報を見つけることができませんでした． 'No relevant information was found for your question.'
	U:	表を作りたい 'I want to make a table'
	S:	以下の選択肢から選んでください． 'Please select the following choices.'
		1. ○ PowerPoint で表を作成する 'Making a table in PowerPoint'
		2. ○ Word で表を作成する 'Making a table in Word'
		3. ○ ...
	evaluation	**success:text**
B-2	U:	強制終了します 'forced termination'
	S:	以下の選択肢から選んでください． 'Please select the following choices.'
		1. ○ EXCEL: 強制終了およびハングアップの際のチェック項目 'EXCEL: check-ups in case of forced termination or hang-up'
		2. ○ EXCEL: 強制終了によって正常に保存されなかったファイルを回復する方法 'EXCEL: how to recover a file which is not normally saved because of forced termination'
		3. ○ ...
	evaluation	**success:text**
B-3	U:	Linux
	S:	以下の選択肢から選んでください． 'Please select the following choices.'
		1. ○ Linux を削除し Windows 2000 または NT をインストールする方法 'How to remove Linux and install Windows 2000 or NT'
		2. ○ Linux RPC クライアントからサービス拒否攻撃を実行できる可能性がある 'Denial-of-Service attack possible from Linux RPC client'
		3. ○ ...
	evaluation	**success:text**
B-4	U:	FreeBSD
	S:	入力された質問から，該当する情報を見つけることができませんでした． 'No relevant information was found for your question.'
	evaluation	**success:no_text**
B-5	U:	自動校正を解除したい 'I want to cancel the spell checker'
	S:	入力された質問から，該当する情報を見つけることができませんでした． 'No relevant information was found for your question.'
	proper text	[WD98] 文章校正に関する Q&A '[WD98] Q&A about the text checker'
	evaluation	**failure:text**
B-6	U:	ホームページの文字が大きい 'Characters on homepages are too large'
	S:	以下の選択肢から選んでください． 'Please select the following choices.'
		1. [HOW TO] 大きな文字配列を初期化する方法'[HOW TO] How to initialize large arrays of strings'
		2. [SNA] 127 よりも大きい文字が NvRunCmd によって間違ってマップ '[SNA] Characters above 127 were incorrectly mapped by NvRunCmd'
		3. ...
	a proper text	表示される Web ページの文字のサイズを変更する 'Change character size on displayed web pages'
	evaluation	**failure:text**
B-7	U:	CAD ソフトが動かない 'A CAD software does not work'
	S:	以下の選択肢から選んでください． 'Please select the following choices.'
		1. [OFFXP] Office のセットアップ後に DVD ソフトが動作しない問題について '[OFFXP] DVD softwares are not worked after Office is set up'
	evaluation	**failure:no_text**

Table 5.4: Evaluations of dialog sessions (session type C)

C-1	U: エラーが発生しました． ‘An error was caused.’ S: エラーはいつ発生しますか? ‘When was the error caused?’ 1. Windows 起動中 ‘while booting Windows’ 2. 印刷時 ‘while printing out’ 3. ... U: Windows 起動中 ‘while booting Windows’ S: あなたがお使いの Windows を選んでください． ‘Which Windows are you using?’ 1. Windows 95 2. Windows 98 3. ... U: Windows 98 S: 以下の選択肢から選んでください．‘Please select the following choices.’ 1. ○ W98:起動時のフォントドライバが読み込めないエラー ‘W98: An error in trying to load font drivers on start-up’ 「JIS フォントドライバーがインストールされていません」等のフォントの読み込みに関するエラーが発生した ‘Font loading errors such as “The JIS font driver has not been installed” were caused’ 2. ○ W98:起動時に「< ファイル名 > が不正かありません」のエラーについて‘W98: About an error “[filename] is illegal or not found” on start-up’ 3. ...
	evaluation **success:text**
C-2	U: XBOX について教えて ‘Tell me about XBOX’ S: Microsoft が開発した次世代家庭用ゲーム機． ‘A next-generation video game machine developed by Microsoft.’ 詳しくは以下のリンクを参照してください． ‘For more details please refer to the following hyperlinks.’ 1. ○ XBOX 2. ○ 発売予定のゲームタイトル ‘game titles to be scheduled to go on sale’ 3. ○ ...
	evaluation **success:text**

Table 5.5: Evaluations of dialog sessions (session type D)

D-1	U: あなたは誰ですか？ ‘Who are you?’ S: Dialog Navigator です． ‘I’m Dialog Navigator.’
	evaluation **misc:proper**
D-2	U: 夏休みの天気が知りたい ‘Please tell me the weather information during the summer holidays.’ S: 入力された質問から，該当する情報を見つけることができませんでした． ‘No relevant information was found for your question.’
	evaluation **misc:proper**
D-3	U: お名前は？ ‘What’s your name?’ S: 以下の選択肢から選んでください． ‘Please select the following choices.’ 1. 用語の説明: 名前 ‘glossary: name’ 2. Access のフィールド，コントロール，およびオブジェクトの名前について ‘About names of fields, controls, and objects in Access’ 3. ...
	evaluation **misc:improper**

Note that the above evaluations exactly grasp behaviors of the system in real situations, in contrast to evaluation based on text collections. However, it is impossible to estimate user intentions exactly: all the users of the system are anonymous, because they can access the system on the web without any registrations. To cope with the problem, we categorized the dialog sessions into the following four session types, and defined a guideline to evaluate dialog sessions for each session type:

- **session type A**:

  Dialog sessions in which user questions were specific, that is, all information for determining relevant texts was specified (Table 5.1 and Table 5.2). Texts that had all of the specified information are judged to be relevant.

  If a session had shown relevant texts as the choices, it is judged to be **success:text** (A-1 and A-2). If not, the judge depends on exhaustive search of the text collections by the subject, using a keyword-based text retrieval system[1]: if relevant texts are found, the session is judged to be **failure:text** (A-4 and A-5); if relevant texts are not found, and the system answered that no relevant texts were found, the session is judged to be **success:no_text** (A-3); if relevant texts are not found, and the system showed other irrelevant texts, the session is judged to be **failure:no_text** (A-6).

  Note that regardless of that a user asked some vague questions at the beginning of a session, the session in which all information for determining relevant texts was specified is categorized into this type (A-2 and A-4).

- **session type B**:

  Dialog sessions in which user questions are vague, that is, some information for determining relevant texts is not specified, excluding sessions in which *dialog cards* are used (Table 5.3). For dialog sessions of this type, it is impossible to judge which texts agreed with user situations. Therefore, texts that had all of the specified information are judged to be relevant.

  If a user question was expressed as only one keyword, every text which has the keyword is judged to be relevant (B-3 and B-4).

- **session type C**:

  Dialog sessions in which *dialog cards* are used (Table 5.4). If user selections reached

[1] A dedicated system which retrieves all matched texts with inputted keywords.

the bottom of the hierarchical structures of the dialog cards, and relevant texts were shown, the session is judged to be **success:text**. It is seldom that no relevant text were shown, because we have continuously checked whether relevant texts were shown for each choice of a dialog card.

- **session type D**:
  Dialog sessions in which user questions are out of the system domain (Table 5.5). If the question was answered with a dialog card (D-1), or if the system answered that no relevant texts were found as a result of matching with the text collections (D-2), the session is judged to be **misc:proper**. If matched irrelevant texts were shown (D-2), the session is judged to be **misc:improper**.

Table 5.1∼5.5 shows examples of evaluations of dialog sessions for each session type. In each table, "U:" indicates an utterance (a question or a choice) of a user, "S:" indicates a response by the system, and "◯" shows a text that is evaluated as a "relevant text" by the subject.

The right side (the column of "sum") of Table 5.6 shows the results of the evaluation of the dialog sessions. The total success ratio to the 230 dialog sessions (excluding **misc** sessions) was 76%.

The left side of Table 5.6 shows the relation between the evaluations of each dialog session and whether were dialog cards used in the session. At the time of the evaluation, there were 216 dialog cards, and those had a hierarchical structure of not more than three levels. The total ratio of dialog sessions in which dialog cards were used to the 230 dialog sessions was 17% (= 38/230), and all of the dialog sessions were successful. In addition, the dialog cards also worked well for out-of-domain (**misc**) user questions that were ranged by those cards. We therefore conclude that the strategy of using the dialog cards generally works well.

Most of the failures of dialog sessions were caused by the lack of the text collections and the synonymous expression dictionary. If there are no relevant texts for a user question in the text collections, it is difficult for the system to answer that it can find no relevant texts, such as A-3; in many cases, it shows irrelevant texts such as A-6 and B-7. Suppose that the parameter t for ignoring texts which had small scores (Table 2.4) were increased, such failures (**failure:no_text**) would be reduced; but in compensation for the reduction, about dialog sessions for which relevant texts existed in the text collections, **success:text**

Table 5.6: Evaluation of dialog sessions, with the usages of dialog cards

evaluation		Whether were dialog cards used? Yes	No	sum
success:	**text**	38 (100%)	111 (58%)	149 (65% / 39%)
	no_text	0 (0%)	25 (13%)	25 (11% / 7%)
	(sum)	38 (100%)	136 (71%)	174 (**76%** / 46%)
failure:	**text**	0 (0%)	15 (8%)	15 (7% / 4%)
	no_text	0 (0%)	41 (21%)	41 (18% / 11%)
	(sum)	0 (0%)	56 (29%)	56 (**24%** / 15%)
subtotal (without **misc**)		38 (100%)	192 (100%)	230 (**100%** / 61%)
misc:	**proper**	57 (——)	0 (——)	57 (—— / 15%)
	improper	3 (——)	88 (——)	91 (—— / 24%)
	(sum)	60 (——)	88 (——)	148 (—— / 39%)
total		98 (——)	280 (——)	378 (—— / 100%)

(unit: # of dialog sessions)

would be reduced, and **failure:text** would be increased. To cope with the trade-off, **failure:text** such as A-5 and B-6 should be reduced by matching relevant texts with user questions by enriching the synonymous expression dictionary.

Some sessions as A-4 failed because of the lack of exploiting dialog contexts. To reduce such failures, matching methods that take dialog contexts into account are needed.

The success ratio at the time of the release of Dialog Navigator was about 60%. To improve the performance, we have continuously modified the synonymous expression dictionary and the dialog cards, by analyzing obvious failures in the dialog database. As a result, the success ratio has increased over 70% until now, as shown in Table 5.6.

5.4.2 Analysis of Behaviors of Users and the System

To verify effectiveness of the architecture of Dialog Navigator, we examined the 378 dialog sessions described in the previous subsection, for how users acted, and for how the system responded to the users (Figure 5.3).

In the 378 dialog sessions, there were 518 user question inputs by keyboards, 19% ($= (32 + 66)/518$) of which were responded by the dialog cards. In addition, an examination of relations between lengths of user questions and the system responses (Table 5.7)

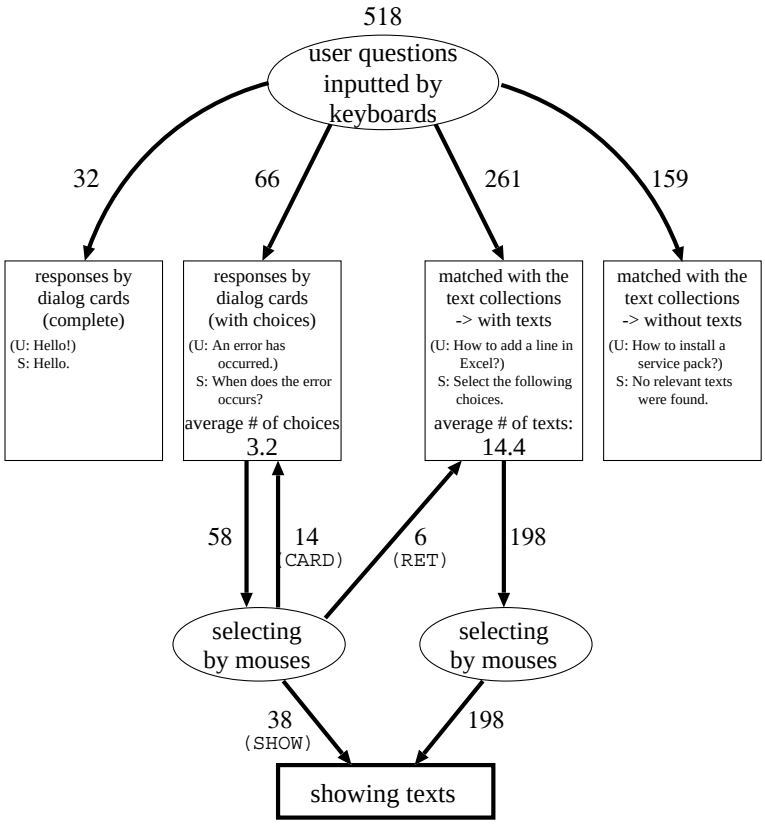


Figure 5.3: Frequency distribution of user actions and system responses

Table 5.7: Lengths of user questions and system responses

# of *bunsetsu* in user questions	responses by dialog cards		matching with text collections		
	complete	with choices	with texts	no texts	total
1	29 (13%)	17 (8%)	115 (52%)	59 (27%)	220 (100%)
2	3 (2%)	37 (28%)	46 (35%)	47 (35%)	133 (100%)
3	( )	10 (14%)	33 (45%)	30 (41%)	73 (100%)
4	( )	2 (6%)	22 (65%)	10 (29%)	34 (100%)
≥ 5	( )	( )	45 (78%)	13 (22%)	58 (100%)
total	32 (6%)	66 (13%)	261 (50%)	159 (31%)	518 (100%)

(unit: frequency)

Table 5.8: Lengths of user questions and matched texts

# of *bunsetsu* in user questions	average # of matched texts	percentages of relevant texts
1 (115 times)	18.2	49%
2 (46 times)	9.1	28%
3 (33 times)	16.0	22%
4 (22 times)	10.5	10%
≥ 5 (45 times)	10.6	11%
total(261 times)	14.4	35%

showed that the dialog cards mainly corresponded to shorter user questions (≤ 3 *bunsetsu*). Typically, the shorter a user question is, the vaguer it is. We can therefore conclude that our strategy of making asking backs for vaguer user questions (the upper part of the triangle in Figure 4.1) by the dialog cards works efficiently.

Another evaluation of relations between lengths of user questions and matching results with the text collections (Table 5.8) showed that percentages of relevant texts became worse as user questions became longer. Typically, the longer a user question is, the more technical it is. Therefore, the text collections may not cover such user questions enough.

As for the relations between lengths of user questions and average numbers of matched texts, the user questions that consisted of only one *bunsetsu* matched especially a lot of texts. It was mainly caused by a lot of one keyword inputs by users, such as B-3 in Table 5.3. By contrast, user questions which had a certain lengths matched appropriate numbers of texts, due to the parameters for limiting the number of choices shown in Table 2.4.

5.4.3 Evaluation of Description Extraction

To examine the results of the description extraction method, we randomly selected 100 user questions that were replied with not less than five choices of matched texts, from the part of the dialog database (1st - 31st August 2002). Next, a subject gave evaluations for each description of the text that was ranked as top five choices, and was extracted from texts of Support KB. In case of the evaluation, we excluded 152 texts the titles or entries of which were selected as the representative sentences (including all of Glossary

and Help texts, and some texts of Support KB in which their titles had the largest score; see Subsection 2.7.2), because each of the texts had the title as a description. As a result, 348 ($= 100 \times 5 - 152$) descriptions were evaluated.

Each extracted description was evaluated from the viewpoint whether the extracted descriptions had helpful, necessary, and successful information for users' choices. Specifically, the subject firstly compared each five choice (description) for a user question with each other, and the subject judged which information was the most important for the user's choice (we call the information as **MII**: Most Important Information). After that, the subject gave either of the following evaluations for each description:

- **proper**:
  The description had MII just enough.
- **insufficient**:
  The description did not have MII enough.
- **verbose**:
  The description had other lots of information than MII. Roughly, the subject gave this evaluation if the number of characters that expressed other information than MII was more than half of the number of characters that expressed MII.

Table 5.9 shows the result of the evaluation on description extraction. The ratio of proper descriptions to the all evaluated descriptions was 61%. Assuming that every title or entry selected as a representative sentence were proper, 73% of extracted descriptions would be proper. In addition, the average length (number of characters) of extracted description was 68.9, and that of original sentences was 81.6. Therefore, the compression ratio of the method was as follows:

$$\left(1 - \frac{\text{the average length of extracted descriptions}}{\text{the average length of original sentences}}\right) \times 100 = \left(1 - \frac{68.9}{81.6}\right) \times 100 = 15.6(\%)$$

Table 5.10 shows examples of the evaluations of the extracted descriptions. In the table, "U:" indicates an utterance (a question or a choice) of a user, "S:" indicates a response by the system. Firstly, the subject judged that "specific environments in which a computer does not sound" as MII. After that, the subject gave either of the evaluations for each extracted descriptions: the descriptions of No. 2 and No. 5 were given **proper**, because they had enough information about types of played sound files; in contrast, the

Table 5.9: Evaluation on description extraction

evaluation	# of choices
proper	213 (61%)
insufficient	27 (8%)
verbose	108 (31%)
total	348 (100%)

Table 5.10: Examples of the evaluations on the extracted descriptions

extracted descriptions	original sentences	evaluation
U: 音が出ない 'No sound'		
S: 以下の選択肢から選んでください.	'Please select the following choices.'	
1. [NT] Crystal Audio や SoundBlaster AWE32 利用時に音が出ない '[NT] No audio with Crystal Audio or SoundBlaster AWE32'	(the title)	
2. コントロールパネルの [サウンド] から CHIMES WAV ファイルをテストした場合、ボリューム設定に関わらず 'If you test the CHIMES.WAV file from the Control Panel Sound applet, regardless of the volume setting'	コントロール パネル の [サウンド] から CHIMES.WAV ファイルを テストした場合、ボリューム設定に関わらず、音は出ません。'If you test the CHIMES.WAV file from the Control Panel Sound applet, regardless of the volume setting, there is no sound.'	**proper**
3. 音楽の再生時に USB スピーカーからポップ音が出る 'When you are listening to audio over USB speakers, you may hear a pop'	(the title)	
4. YAMAHA YSTMS55D USB スピーカセットのインストール後、スピーカのボリュームコントロールノブを使っても、非常に音が小さい、または、音が出ない 'After you install the Yamaha YSTMS55D USB speaker set, the speakers have extremely little or no volume when you attempt to use the volume control knob on the speakers.'	YAMAHA YSTMS55D USB スピーカ セットのインストール後、スピーカのボリューム コントロール ノブを使っても、非常に音が小さい、または、音が出ないことがあります。'After you install the Yamaha YSTMS55D USB speaker set, the speakers may have extremely little or no volume when you attempt to use the volume control knob on the speakers.'	**verbose**
5. Windowsサウンド (.WAV) ファイルを再生時に 'When you play a Windows Sound (.wav) file'	Windows サウンド (.WAV) ファイルを再生時に、音が出ない。'When you play a Windows Sound (.wav) file, you hear static.'	**proper**

description of No. 4 were given **verbose**, because it had extra information such as other conditions, although it had the name of the sound device as the environment.

The relatively small compression ration of the proposed method was mainly caused by the high ratio of **verbose** descriptions. Specifically, the method often left extra segments which had no important information for users' choices, such as in the No. 4 description of Table 5.10. To improve the result, the system should recognize important segments based on mutual comparison of all choices, and it should exclude other segments. In addition, recognition of coordination structures is required. As for the No. 4 description of Table 5.10, both "非常に音が小さい" (*hijō-ni oto-ga chīsai*, extremely little volume) and "音が出ない" (*oto-ga de-nai*, no volume) should be removed: those two clauses form coordination, and both clauses are verbose for asking-backs.

Most of **insufficient** extracted descriptions were caused by unsatisfactory selections of representative sentences, each of which was the original sentence for the description. Such original sentences did not represent entire texts. To get proper descriptions, the matching methods of user questions with texts should be improved. However, the limitation that only one representative sentence for each text often caused unsatisfactory descriptions. For example, our method did not work well if each of separate two sentences had encountered problems (*e.g.* "an error has occurred") or specific situations (*e.g.* displayed error messages) respectively. To cope with the problem, sophisticated linguistic analysis such as discourse analysis is required.

5.5 Summary of this Chapter

This chapter described the real-world application of our proposed methods on the practical dialog system *Dialog Navigator*, and showed the results and effectiveness of our methods.

To evaluate our methods in the real world, we proposed specific user interface and system architecture to realize the asking-backs strategy described in Chapter 4, and several devices to get a lot of user accesses by improving the usability of the system. The evaluation of the system consisted of three types of examinations. The first examination was based on the dialog session units, and showed that 76% of the dialog sessions were successful. The second examination analyzed the behaviors of the users and the system, and validated the strategy for clarifying vague questions based on the two types of asking-backs, *i.e.,* description extraction and dialog cards. The last examination evaluated the

outputs of the description extraction, and showed that 61% of extracted descriptions were proper.

Chapter 6

Conclusion

When we have to find answers for our questions on using complicated instruments without aids of others, various gaps between the questions and the answers prevent us from reaching the answers. Although a lot of previous studies on information retrieval have tackled this problem, some of the gaps have not been satisfactorily solved. This thesis proposed a solution to resolve such gaps, based on real-world text collections. The solution consists of two strategies: precise and flexible matching of user questions with texts, and two kinds of asking-backs based on the matching. To evaluate the solution, we implemented the proposed solution on a dialog system, *Dialog Navigator*, based on real-world text collections.

This chapter demonstrates the author's contributions toward realization of a system which acts as experts or call centers, by summarizing the author's achievements. Finally, this chapter concludes with the future directions of this study.

6.1 Contributions

Chapter 1 showed a survey of previous studies on information retrieval firstly, and a detailed examination of question logs collected by an existing text retrieval system. The examination indicated that there are some types of gaps between user questions and answers. The gaps mainly consist of expression gaps and vagueness gaps. Methods of matching user questions with texts are required for resolving the expression gaps, and methods of interacting with users are required for resolving the vagueness gaps and the belief gaps. Moreover, these methods should be applicable to large text collections. However, almost all of the previous studies do not satisfy either of these requirements. To satisfy all of the requirements, this chapter proposed a solution based on precise and

flexible matching methods as application of natural language processing techniques: the expression gaps are resolved by the matching methods themselves; and the vagueness gaps are resolved by two kinds of asking-backs based on the matching methods.

Chapter 2 described the methods which realize the precise and flexible matching of user question with texts. To satisfy the requirements for achieving the goal of this thesis, the methods based on full-parsing results of user questions and texts were developed. To be exact, all these methods were founded on sentence stricture analysis using a robust fairly accurate Japanese parser KNP: expression gaps are resolved using synonymous expression dictionary and ontological dictionary, in consideration of modifier-head relations; and the similarity score between user questions and texts is calculated based on analyzed sentence structures, giving large points to matches of modifier-head relations. In addition, several devices to improve the performance were proposed: question type estimation for selecting text collections, text selection by product names, removal of verbose final expressions, assignment of negation flags, and indexing for quick retrieval of texts.

An evaluation on these methods using testsets demonstrated that the weighting on modifier-head relations significantly improved the performance, and other devices were also effective. This chapter also discussed the matching failures in the testsets, and indicated that further advancements of natural language processing techniques would be required.

Finally, this chapter compared the author's methods with previous studies on information retrieval. The comparison focused on full-parsing-based matching indicated that the author's methods are an extension of methods of open-domain question answering systems, rather than that of text retrieval systems. Another comparison focused on resolving expression gaps showed that the author's methods have important advantages over other methods: capability of resolving phrase-level gaps, and applicability to large text collections.

Chapter 3 extended the methods described in the previous chapter toward processing metonymy. Metonymy is a figure of speech in which the name of one thing is substituted for that of something to which it is related, and it appears frequently in both user questions and texts. It is critical for the goal of this thesis to process metonymy, because metonymy often causes gaps of modifier-head relations between user questions and texts.

To solve the problem, this chapter proposed a method for automatic acquisition of pairs of metonymic expressions and their interpretative expressions, and a method for using the acquired pairs to resolve the gaps of modifier-head relations. Firstly, the acquisition method was applied to corpora which consisted of a lot of user questions and text collections, and 1,126 pairs of metonymic expressions and their interpretative expressions were derived. An evaluation showed that more than 80% of the pairs were correct. Another evaluation on the effectiveness of introducing the acquired pairs into the matching methods descried in Chapter 2 using testsets demonstrated significant improvements over the baseline methods.

Finally, this chapter showed the advantage of the author's method over other ones employed by previous studies on processing metonymy: this method works without manually constructed knowledge structures, which require heavy cost of construction and maintenance; and this method treats both recognition and interpretation process of metonymy, although the recognition process has been left by the previous studies.

Chapter 4 tackled the resolution of another important type of the gaps, *i.e.,* the vagueness gaps, by exploiting the matching methods. Firstly, a preliminary examination of question logs of a natural language based text retrieval system showed that about 30% of the user questions were vague, and matches of such questions with texts resulted in a pile of matched texts. Based on this fact, this chapter emphasized that clarification of such vague questions is required. To realize that, this chapter proposed a strategy of making two types of asking-backs complementarily: *description extraction* as a bottom-up approach, and *dialog cards* as a top-down approach.

The former approach works based on the fact that the neighborhoods of the part that match user questions describe specific symptoms and conditions of problems which users often encounter. The extraction of the neighborhoods usually helps users to find relevant texts. Note that the extraction would not be possible without the precise and flexible matching methods proposed in the previous chapters.

However, if a user question is too vague, the extraction often does not work, because such vague question will match too many texts. The latter approach, *i.e.,* the dialog cards, covers the inadequacy of the description extraction. The dialog cards were constructed based on the manual for the operators of the call center in Microsoft Corporation, and systematized procedures for interactively clarifying frequently asked vague questions. The

matching methods in the previous chapters are also helpful in selecting appropriate dialog cards.

Finally, this chapter compared the author's strategy with those of previous studies, and showed the advantages over them. Generally, asking-back methods employed by text retrieval systems are not suitable for the resolution of the vagueness gaps; artificial-intelligence-based approaches have no scalability; the dialog help system of Kyoto University are based on knowledge base written in natural language, but the knowledge base should be described using uniform and limited expressions. In contrast, the author's strategy realizes asking-backs for vague questions, based on real-world large text collections.

Chapter 5 described the real-world operation of the methods described in the previous chapters, and evaluations of the results. Firstly, this chapter pointed out that it is inevitable for methods of resolving the vagueness gaps to be evaluated by motivated users in the real world. To perform the evaluation, all the methods were implemented on a dialog system based on the real-world text collections provided by Microsoft Corporation: *Dialog Navigator*, which targets ordinary users of Windows operating systems on personal computers. To be exact, a specific architecture which realizes the methods proposed in this thesis and a user interface which improves usability were devised. The service of the system was started on the website of Microsoft Corporation in April 2002, and all conversation logs of user questions and system responses have been stored as a dialog database.

Based on the database, three types of examinations were performed. The first examination evaluated each dialog session, and showed that 76% of the dialog sessions were successful. The result suggests that the matching methods work well overall. The second examination analyzed the behaviors of the users and the system, and validated the proposed strategy for clarifying vague questions based on the two types of asking-backs, *i.e.,* description extraction and dialog cards. The last examination evaluated the outputs of the description extraction, and showed that 61% of extracted descriptions were proper.

Now let us summarize the contributions of this study. This thesis proposed a concrete solution to cope with circumstance surrounding users of recent complicated instruments: the users often have questions on using the instruments, and large text collections for answering the questions have been gathers; however, various gaps between the questions

and texts, chiefly expression gaps and vagueness gaps, prevent us from reaching appropriate texts. Previous studies on information retrieval had troubles with resolving the gaps: some studies (*e.g.* ask-backs by keywords, texts, or clusters based on conventional keyword-based text retrieval systems) were insufficient to resolve the vagueness gaps, and other studies (*e.g.* application of artificial intelligence techniques) were not applicable to large text collections. In contrast, this study proposed a solution of combining the following two methods: resolution of the expression gaps by flexible and precise matching methods of user questions with texts; and resolution of the vagueness gaps by making asking-backs based on the precise and flexible matching methods.

6.2 Future Directions

This thesis proposed a solution to resolve various gaps between user questions and their answers. However, there are still gaps to be resolved. For example, some of expression gaps beyond keyword level are left as future issues. Resolution of vagueness gaps is the main topic of this thesis, but it is still required to be improved. Resolution of belief gaps are left behind, although it will be critical for novice users. Moreover, it is a big problem that the author's methods depend on some manually constructed knowledge structures, such as the synonymous expression dictionary and the dialog cards. The costs of constructing them are relatively small than that of previous studies, but are still large when we construct the knowledge structures which cover targeted domains sufficiently. In addition, the keyboard-based user interface requires users to pay a large amount of labor for rephrasing their questions, although rephrasing is vital for efficient dialogs between users and the system. The author thinks that a speech interface is suitable for rephrasing.

To cope with the above issues, the author would like to study for the following directions, based on recent developments in natural language processing techniques:

- Further improvements of expression gap resolution.

  As mentioned at the end of Section 1.2, some of expression gaps beyond sentence level, including anaphora and ellipses, are left as future issues, even though the resolution of such gaps is essential for contextual interpretations of user questions [26]. The anaphora and ellipses resolution requires case frames which cover the diversity of verb usages. In the past, the coverage has been unsatisfactory, because the case frames have been constructed manually. However, automatic construction of case

frames, and its application to the anaphora and ellipses resolution are becoming available [53]. The author would like to tackle the contextual interpretations of user questions, based on the techniques, and examination of the question logs collected by Dialog Navigator.

Application of *paraphrasing* techniques to resolve the expression gaps will be another topic. The coverage of the synonymous expression dictionary is currently bound by the cost of constructing it. It is desirable that paraphrasing techniques which are applicable to large text collections are developed.

- Further improvements of vagueness gap resolution.

  Construction of the dialog card collection which sufficiently covers vague user questions requires a large cost. The author would like to improve the method of description extraction, by introducing generalization of similar descriptions among matched texts.

- User modeling.

  To cope with the belief gaps, a model which simulates users' beliefs is required. For example, if a user makes a question based on incorrect beliefs, the system should make asking-backs which correct the beliefs, by detecting inconsistency of the beliefs with answer texts using the model.

  Interactive instructions of answer texts based on a user model are also important. This thesis covers only the process of finding answer texts, and leaves the instruction process, which is another important role of call centers and experts. To realize the instruction process, user modeling is also required, because favorable ways of the instructions could vary with experience, skills, or circumstances of users. For example, novice users may prefer to be told each action step by step. By contrast, experts may favor brief instructions.

- Speech interface.

  This thesis emphasized the importance of interactions between users and systems in resolving the various gaps between questions and answers. The author thinks that introduction of a speech interface will promote the interactions, because rephrasing on the speech interface is less difficult than that on the keyboard-based interface.

Currently, Dialog Navigator has been extended for speech input [54]. The most critical issue for the speech interface is recognition errors. To cope with the issue, two criteria of making asking-backs for fixing the errors were introduced. The service of speech-based Dialog Navigator was started on April 2004. The author would like to improve the system by examining the collected user questions.

Bibliography

[1] H. P. Luhn. The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development*, (2):159–165, 1958.

[2] Panel Chairman W. O. Baker. *Improving the Availability of Scientific and Technical Information in the United States*. President's Science Advisory Committee, 1958.

[3] Ruth Atwood. Grass-Roots Look at MEDLARS. *Bull Med Libr Assoc.*, 52(4):645–651, 1964.

[4] G. Salton, editor. *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall, 1971.

[5] NIST and DARPA. *The Twelfth Text REtrieval Conference (TREC 2003)*. NIST Special Publication SP 500-255. 2003.

[6] Carol Peters, Martin Braschler, Julio Gonzalo, and Michael Kluck, editors. *Advances in Cross-Language Information Retrieval*, Lecture Notes in Computer Science 2785. Third Workshop of the Cross-Language Evaluation Forum, CLEF 2002, Rome, Italy, Springer, 2003. Revised papers.

[7] Keizo Oyama, Emi Ishida, and Noriko Kando, editors. *NTCIR Workshop3: Proceedings of the Third NTCIR Workshop on Research in Information Retrieval, Automatic Text Summarization and Question Answering*, Tokyo, Japan, 2003. National Institute of Informatics.

[8] K. Hammond, R. Burke, C. Martin, and S. Lytinen. FAQ Finder: A Case-Based Approach to Knowledge Navigation. In *Proceedings of the 11th Conference on Artificial Intelligence for Applications*, 1995.

[9] Jun'ichi Fukumoto, Tsuneaki Kato, and Fumito Masui. Question Answering Challenge (QAC-1) Question answering evaluation at NTCIR Workshop 3. In *Working Notes of the Third NTCIR Workshop Meeting, Part IV: Question Answering Challenge (QAC1)*, pages 1–10. National Institute of Informatics, 2002.

[10] G. Salton. Automatic Text Analysis. *Science*, 168:335–343, 1970.

[11] Julian Kupiec. MURAX: A robust linguistic approach for question answering using an on-line encyclopedia. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 181–190, 1993.

[12] Sanda Harabagiu, Dan Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Răzvan Bunescu, Roxana Gîrju, Vasile Rus, and Paul Morărescu. The role of lexico-semantic feedback in open-domain textual question-answering. In *Proceedings of the Association for Computational Linguistics*, 2001.

[13] Livier Ferret, Brigitte Grau, Martine Hurault Plantet, Gabriel Illouz, Christian Jacquemin, Nicolas Masson, and Paule Lecuyer. QALC: the Question-Answering System of LIMSI-CNRS. In *The Ninth Text REtrieval Conference (TREC-9)*, pages 235–244. NIST Special Publication, 2001.

[14] Masaki Murata, Masao Utiyama, and Hitoshi Isahara. A Question-Answering System Using Unit Estimation and Probabilistic Near-Terms IR. In *Working Notes of the Third NTCIR Workshop Meeting, Part IV: Question Answering Challenge (QAC1)*, pages 47–54. National Institute of Informatics, 2002.

[15] Robert S. Taylor. Question-Negotiation and Information Seeking in Libraries. *College and Research Libraries*, 29(3):178–194, 1968.

[16] Lucy A. Suchman. *Plans and Situated Actions: The Problem of Human-Machine Communication*. Cambridge University Press, New York, 1987.

[17] D.Sadek. Design Consideration on Dialogue Systems: From Theory to Technology - The Case of Artimis -. In *Proc. ESCA workshop on Interactive Dialogue in Multi-Modal Systems*, pages 173–187, 1999.

[18] J. Rocchio. Relevance Feedback in Information Retrieval. In G. Salton, editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*, chapter 14, pages 313–323. Prentice-Hall, 1971.

[19] Marti A. Hearst and Jan O. Pedersen. Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results. In *Proceedings of ACM SIGIR*, 1996.

[20] Krista Lagus. *Text Mining with the WEBSOM.* Number 110 in Acta Polytechnica Scandinavica, Mathematics and Computing Series. D.Sc.(Tech) Thesis, Helsinki University of Technology, Espoo, Finland, 2000.

[21] Robert Wilensky, Yigal Arens, and David Chin. Talking to UNIX in English: An Overview of UC. *Communications of the ACM*, 27(6):574–593, 1984.

[22] Karen Sparck Jones. *Automatic keyword classification for information retrieval.* Butterworths, London, 1971.

[23] Yonggang Qiu and Hans-Peter Frei. Concept based query expansion. In *Proceedings of SIGIR '93*, pages 160–169, 1993.

[24] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[25] Sanda M. Harabagiu. Deriving Metonymic Coercions from WordNet. In *Workshop on Usage of WordNet in Natural Language Systems, COLING-ACL '98*, 1998.

[26] Sadao Kurohashi and Wataru Higasa. Dialogue Helpsystem based on Flexible Matching of User Query with Natural Language Knowledge Base. In *Proceedings of 1st ACL SIGdial Workshop on Discourse and Dialogue*, pages 141–149, Hong Kong, 2000.

[27] Sadao Kurohashi and Makoto Nagao. A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, 20(4), 1994.

[28] George Lakoff and Mark Johnson. *Metaphors we live by.* University of Chicago Press, 1980.

[29] T. Strzalkowski and J. Carballo. Recent Developments in Natural Language Text Retrieval. In *The Second Text REtrieval Conference (TREC-2)*, pages 123–136. NIST Special Publication, 1993.

[30] T. Strzalkowski, F. Lin, J. Wang, L. Guthrie, J. Leistensnider, J. Wilding, J. Karlgren, T. Straszheim, and J. Perez-Carballo. Natural Language Information Retrieval. In *The Fifth Text REtrieval Conference (TREC-5)*, pages 291–314. NIST Special Publication, 1996.

[31] C. Zhai, X. Tong, N. Milic-Frayling, and D.A. Evans. Evaluation of Syntactic Phrase Indexing – CLARIT NLP Track Report. In *The Fifth Text REtrieval Conference (TREC-5)*, pages 347–358. NIST Special Publication, 1996.

[32] T. Strzalkowski and K. Sparck Jones. NLP Track at TREC-5. In *The Fifth Text REtrieval Conference (TREC-5)*, pages 97–102. NIST Special Publication, 1996.

[33] Sadao Kurohashi and Makoto Nagao. *Japanese Morphological Analysis System JUMAN version 3.61 Users Manual*. Graduate School of Informatics, Kyoto University, http://www-lab25.kuee.kyoto-u.ac.jp/nl-resource/juman.html, 1999. (in Japanese).

[34] Daisuke Kawahara, Nobuhiro Kaji, and Sadao Kurohashi. Question and Answering System based on Predicate-Argument Matching. In *Working Notes of the Third NTCIR Workshop Meeting, Part IV: Question Answering Challenge (QAC1)*, pages 21–24. National Institute of Informatics, 2002.

[35] Peter Mark Roget. *Thesaurus of English words and phrases : classified and arranged so as to facilitate the expression of ideas and assist in literary composition*. John B. Alden, New York, 1852. Enlarged and improved, partly from the author's notes, and with a full index by John Lewis Roget.

[36] Christiane Fellbaum, editor. *WordNet: an electronic lexical database*. The MIT Press, 1998.

[37] Sadao Kurohashi and Yasuyuki Sakai. *Nihongo Hyōgen no Jūnan na Shōgō* (Flexible Matching of Expressions in Japanese). In *Proceedings of The Seventh Annual Meeting of The Association for Natural Language Processing*, pages 343–346, Tokyo, Japan, 2001. (in Japanese).

[38] Satoru Ikehara, Masahiro Miyazaki, Satoshi Shirai, Akio Yokoo, Hiromi Nakaiwa, Kentaro Ogura, Yoshifumi Ooyama, and Yoshihiko Hayashi, editors. *Nihongo Goi Taikei (Japanese Lexical Dictionary)*. NTT Communication Science Laboratories, Iwanami Shoten, 1997. (in Japanese).

[39] Tetsuro Takahashi, Kozo Nawata, Kentaro Inui, and Yuji Matsumoto. Effects of Structural Matching and Paraphrasing in Question Answering. *IEICE Transactions on Information and Systems*, E86-D(9):1677–1685, 2003.

[40] Nobuhiro Kaji and Sadao Kurohashi. Recognition and Paraphrasing of Periphrastic and Overlapping Verb Phrases. In *Proceedings of Workshop on Methodologies & Evaluation of Multiword Units in Real-world Applications (MEMURA2004)*, pages 24–30, 2004.

[41] Shin'ichiro Kamei and Takahiro Wakao. Metonymy; reassessment, survey of acceptability, and its treatment in a machine translation system. In *Proceedings of 30th Annual Meeting of the Association for Computational Linguistics (ACL92)*, pages 309–311, 1992.

[42] Dan Fass. met*: A Method for Discriminating Metonymy and Metaphor by Computer. *Computational Linguistics*, 17(1):49–90, 1991.

[43] David Stallard. Two Kinds of Metonymy. In *Proceedings of 31st Annual Meeting of the Association for Computational Linguistics (ACL93)*, pages 87–94, 1993.

[44] Masaki Murata, Qing Ma, Atsumu Yamamoto, and Hitoshi Isahara. Metonymy Interpretation Using X NO Y Examples. In *Proceedings of The 4th Symposium on Natural Language Processing 2000 (SNLP 2000)*, 2000.

[45] Masao Utiyama, Masaki Murata, and Hitoshi Isahara. A Statistical Approach to the Processing of Metonymy. In *Proceedings of The 18th International Conference on Computational Linguistics (COLING 2000)*, pages 885–891, 2000.

[46] Maria Lapata and Alex Lascarides. A Probabilistic Account of Logical Metonymy. *Computational Linguistics*, 29(2):261–315, 2003.

[47] Dan Fass. *Processing metonymy and metaphor*, volume 1 of *Contemporary studies in cognitive science and technology*. Ablex Pub. Corp., 1997.

[48] Wim Peters. Metonymy as a Cross-lingual Phenomenon. In *Proceedings of The ACL 2003 Workshop on the Lexicon and Figurative Language*, pages 1–9, 2003.

[49] Piek Vossen, editor. *EuroWordNet: a multilingual database with lexical semantic networks*. Kluwer Academic, 1998.

[50] Hideki Nishimura, Hiroyuki Kawano, and Toshiharu Hasegawa. Implementation and evaluation of WWW search system RCAAU. In *IEICE technical report. Data engineering (DE) 96-54*, pages 1–6, 1996. (in Japanese).

[51] Akihiko Takano, Yoshiki Niwa, Shingo Nishioka, Makoto Iwayama, Toru Hisamitsu, Osamu Imaichi, and Hirofumi Sakurai. Associate Information Access Using DualNAVI. In *Proceedings of Kyoto International Conference on Digital Libraries 2000 (ICDL '00)*, pages 285–289, 2000.

[52] R. N. Oddy. Information retrieval through man-machine dialogue. *Journal of Documentation*, 33(1):1–14, 1977.

[53] Daisuke Kawahara and Sadao Kurohashi. Zero Pronoun Resolution based on Automatically Constructed Case Frames and Structural Preference of Antecedents. In *Proceedings of The First International Joint Conference on Natural Language Processing (IJCNLP-04)*, pages 334–341, 2004. Hainan Island, China.

[54] Yoji Kiyota, Sadao Kurohashi, Teruhisa Misu, Kazunori Komatani, Tatsuya Kawahara, and Fuyuko Kido. Dialog Navigator: A Spoken Dialog Q-A System based on Large Text Knowledge Base. In *the Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003 Interactive Posters/Demonstrations Sessions)*, pages 149–152, 2003. Sapporo, Japan.

List of Publications by the Author

Major Publications

[1] Yoji Kiyota and Sadao Kurohashi. Automatic Summarization of Japanese Sentences and its Application to a WWW KWIC Index. In *Proceedings of 2001 Symposium on Applications and the Internet (SAINT 2001)*, pages 120–127, 2001. San Diego, USA.

[2] Yoji Kiyota, Sadao Kurohashi, and Fuyuko Kido. Dialog Navigator: A Question Answering System based on Large Text Knowledge Base. In *Proceedings of The 19th International Conference on Computational Linguistics (COLING 2002)*, pages 460–466, 2002. Taipei, Taiwan.

[3] Yoji Kiyota, Sadao Kurohashi, Teruhisa Misu, Kazunori Komatani, Tatsuya Kawahara, and Fuyuko Kido. Dialog Navigator: A Spoken Dialog Q-A System based on Large Text Knowledge Base. In *the Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003 Interactive Posters/Demonstrations Sessions)*, pages 149–152, 2003. Sapporo, Japan.

[4] Yoji Kiyota, Sadao Kurohashi, and Fuyuko Kido. Dialog Navigator: A Question Answering System based on Large Text Knowledge Base. *Journal of Natural Language Processing*, 10(4):145–175, 2003. (in Japanese).

[5] Yoji Kiyota, Sadao Kurohashi, and Fuyuko Kido. Resolution of Modifier-head Relation Gaps using Automatically Extracted Metonymic Expressions. In *Proceedings of The First International Joint Conference on Natural Language Processing (IJCNLP-04)*, pages 171–176, 2004. Hainan Island, China.

[6] Yoji Kiyota, Sadao Kurohashi, and Fuyuko Kido. Resolution of Modifier-head Relation Gaps using Automatically Extracted Metonymic Expressions. *Journal of Natural Language Processing*, 11(4), 2004. (in Japanese, to appear).

Other Publications

[1] Yoji Kiyota, Sadao Kurohashi, Jun'ichi Nakamura, and Makoto Nagao. A Clustering System for Electronic News Articles using Sentence Structure. In *IPSJ SIG Notes, NL-126-11*, pages 77–84, 1998. Kyoto, Japan (in Japanese).

[2] Yoji Kiyota, Sadao Kurohashi, Jun'ichi Nakamura, and Taku Kudo. A Text Retrieval System based on Term Co-occurrence Weighting. In *Proceedings of IREX Workshop*, pages 53–56. IREX Committee, 1999. Tokyo, Japan (in Japanese).

[3] Yoji Kiyota and Sadao Kurohashi. Automatic Summarization of WWW Texts and its Application to a WWW KWIC Index. In *IPSJ SIG Notes, NL-137-5*, pages 31–38, 2000. Kanagawa, Japan (in Japanese).

[4] Yoji Kiyota and Sadao Kurohashi. Dialogue Helpsystem at CIMS, Kyoto University and Automatic Reference Service System at Kyoto University Library. In *IPSJ SIG Notes, NL-137-14*, page 92, 2000. Kanagawa, Japan (in Japanese).

[5] Kazunori Komatani, Tatsuya Kawahara, Yoji Kiyota, Sadao Kurohashi, and Pascale Fung. Restaurant Search System with Speech Interface using Flexible Language Model and Matching. In *IPSJ SIG Notes, SLP-39-30*, pages 177–182, 2001. Tokyo, Japan (in Japanese).

[6] Yoji Kiyota, Sadao Kurohashi, and Fuyuko Kido. Dialog Navigator: A Question Answering System based on Large Text Knowledge Base. In *Proceedings of The Eighth Annual Meeting of The Association for Natural Language Processing*, pages 271–274, 2002. Kyoto, Japan (in Japanese).

[7] Yoji Kiyota, Sadao Kurohashi, and Fuyuko Kido. Resolution of Modifier-head Relation Gaps using Automatically Extracted Metonymic Expressions. In *Proceedings of The Tenth Annual Meeting of The Association for Natural Language Processing*, pages 305–308, 2004. Tokyo, Japan (in Japanese).

[8] Teruhisa Misu, Kazunori Komatani, Yoji Kiyota, Tatsuya Kawahara, and Fuyuko Kido. -Speech Dialog Navigator- Large Scale Document Retrieval System with Spoken Dialog. In *IPSJ SIG Notes, SLP-52-4*, pages 21–26, 2004. Hokkaido, Japan (in Japanese).